Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques

17th International Workshop, APPROX 2014, and
18th International Workshop, RANDOM 2014
September 4–6, 2014, Barcelona, Spain

Edited by

Klaus Jansen
José D. P. Rolim
Nikhil R. Devanur
Cristopher Moore

LIPIcs – Vol. 28 – APPROX/RANDOM'14 www.dagstuhl.de/lipics

Editors

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

José D. P. Rolim
University of Geneva
Geneva
Jose.Rolim@unige.ch

Nikhil R. Devanur
Microsoft Research
Redmond
nikdev@microsoft.com

Cristopher Moore
Santa Fe Institute
New Mexico
moore@santafe.edu

ACM Classification 1998
C.2.1 Network Architecture and Design, C.2.2 Computer-communication, E.4 Coding and Information Theory, F. Theory of Computation, F.1.0 Computation by Abstract Devices, F.1.1 Models of Computation – relations between models, F.1.2 Modes of Computation, F.1.3 Complexity Measures and Classes, F.2.0 Analysis of Algorithms and Problem Complexity, F.2.1 Numerical Algorithms and Problems, F.2.2 Nonnumerical Algorithms and Problems G.1.2 Approximation, G.1.6 Optimization, G.2 Discrete Mathematics, G.2.1 Combinatorics, G.2.2 Graph Theory, G.3 Probability and Statistics, I.1.2 Algorithms, J.4 Computer Applications – Social and Behavioral Sciences

ISBN 978-3-939897-74-3

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/978-3-939897-74-3.

Publication date
September, 2014

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.2014.i

ISBN 978-3-939897-74-3 **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

www.dagstuhl.de/lipics

Contents

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore
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 17th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2014) and the 18th International Workshop on Randomization and Computation (RANDOM 2014), which took place concurrently in Universitat Politècnica de Catalunya Barcelona, Spain, during September 4–6, 2014.

APPROX focuses on algorithmic and complexity issues surrounding the development of efficient approximate solutions to computationally difficult problems, and was the 17th 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), and Princeton (2011), Berkeley (2013). RANDOM is concerned with applications of randomness to computational and combinatorial problems, and was the 18th 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).

Topics of interest for APPROX and RANDOM are: design and analysis of approximation algorithms, hardness of approximation, small space algorithms, sub-linear time algorithms, streaming algorithms, embeddings and metric geometry, mathematical programming methods, combinatorial problems in graphs and networks, game theory, markets and economic applications, geometric problems, packing, covering, scheduling, approximate learning, design and analysis of online algorithms, 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, property testing, phase transitions, computational learning theory, and other applications of approximation and randomness.

The volume contains 31 contributed papers, selected by the APPROX Program Committee out of 64 submissions, and 30 contributed papers, selected by the RANDOM Program Committee out of 62 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 support from the Microsoft Research, USA, the Institute of Computer Science of the Christian-Albrechts-Universität zu Kiel, the Santa Fe Institute, USA, and the Department of Computer Science of the University of Geneva.

September 2014

Nikhil R. Devanur
Klaus Jansen
Cristopher Moore
José D. P. Rolim

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Organization

Program Committees

APPROX 2014

Niv Buchbinder	Tel Aviv University, Israel
Deeparnab Chakrabarty	Microsoft Research, India
Siu On Chan	Microsoft Research UC, Berkeley, USA
Shuchi Chawla	University of Washington, USA
Eden Chlamtac	Princeton University, USA
Nikhil R. Devanur (chair)	Microsoft Research, Redmond, USA
Alina Ene	Princeton University, USA
Konstantinos Georgiu	University of Waterloo, Canada
Telikepalli Kavitha	Indian Institute of Science, Bangalore, India
Ken Ichi Kawarabayashi	National Institute of Informatics, Tokyo, Japan
Jochen Koenemann	University of Waterloo, Canada
Amit Kumar	Indian Institute of Technology, New Delhi, India
Konstantin Makarychev	Microsoft Research, Redmond, USA
Debmalya Panigrahi	Duke University, USA
Thomas Rothvoss	Massachusetts Institute of Technology, USA
Barna Saha	AT&T Shannon Research Laboratory, New Jersey, USA
Bruce Shepherd	McGill University Montreal, Canada
Aravind Srinvasan	University of Maryland, USA
David Williamson	Cornell University, USA

RANDOM 2014

Louigi Addario-Berry	McGill University, Montreal, Canada
Nayantara Bhatnagar	University of Delaware, Newark, USA
Amin Coja-Oghlan	Goethe University, Frankfurt, Germany
David Galvin	University of Notre Dame, South Bend, USA
Valentine Kabanets	Simon Fraser University, Burnaby, Canada
Michael Molloy	University of Toronto, Canada
Cristopher Moore (chair)	Santa Fe Institute, New Mexico, USA
Assaf Naor	New York University, USA
Krzysztof Onak	IBM T.J. Watson Research Center, USA
Dana Ron	Tel-Aviv University, Israel
Alex Russell	University of Connecticut, USA
Dominik Scheder	Tsinghua University, Beijing, China
Devavrat Shah	Laboratory for Information and Decision Systems, Cambridge, USA
Perla Sousi	University of Cambridge, USA
Mario Szegedy	University of New Jersey, Piscataway, USA
Amnon Ta-Shma	Tel-Aviv University, Israel
Thomas Vidick	California Institute of Technology, USA

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

External Reviewers

Dimitris Achlioptas
Susanne Albers
Ammar Ammar
Alexandr Andoni
Per Austrin
Yossi Azar
Eric Bach
Paul Balister
Victor Bapst
Alexander Barg
Surender Baswana
Mohammad Bavarian
Shankar Bhamidi
Umang Bhaskar
Arnab Bhattacharyya
Pritam Bhattacharya
Eric Blais
Milan Bradonjic
Fernando Brandao
Mark Braverman
Guy Bresler
Joshua Brody
Nicolas Broutin
Jaroslaw Byrka
Clément Canonne
Amit Chakrabarti
Siu On Chan
Sourav Chatterjee
Arkadev Chattopadhyay
Po-An Chen
Shahar Chen
Sixia Chen
Mahdi Cheraghchi
Flavio Chierichetti
George Christodoulou
Marek Chrobak
Fabian Chudak
Ilan Cohen
Gil Cohen
Colin Cooper
Artur Czumaj
Olivier Durand de Gevigney
Ronald de Wolf
Dean Doron
Martin Dyer
Klim Efremenko
Charilaos Efthymiou
Leah Epstein
Omid Etesami
Uriel Feige
Sándor Fekete
Moran Feldman
Andreas Feldmann
Nikolaos Fountoulakis
Ariel Gabizon
Shirshendu Ganguly
Ankit Garg
Naveen Garg
Efraim Gelman
Balázs Gerencsér
George Giakkoupis
Oded Goldreich
Simon Griffiths
Elena Grigorescu
Tom Gur
Ori Gurel-Gurevich
Pooya Hatami
Thomas Hayes
Timon Hertli
Samuel Hetterich
Martin Hildebrand
Chien-Chung Huang
Sungjin Im
Yoichi Iwata
Ragesh Jaiswal
Sune Jakobsen
Mark Jerrum
Volker Kaibel
Sagar Kale
Satyen Kale
Matthew Katz
Tali Kaufman
Muhammad Khan
Julia Komjathy
Swastik Kopparty
Ravishankar Krishnaswamy
Florent Krzakala
Janardhan Kulkarni
Nirman Kumar
Ravi Kumar

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Oded Lachish
Kevin Leckey
Marc Lelarge
Virginie Lerays
Vahid Liaghat
Ricardo Restrepo Lopez
Shachar Lovett
Pinyan Lu
Eyal Lubetzky
Gabor Lugosi
Takanori Maehara
Yury Makarychev
Arie Matsliah
Kevin Matulef
Andrew McGregor
Or Meir
Julian Mestre
Ankur Moitra
Sonoko Moriyama
Ben Morris
Elchanan Mossel
Dhruv Mubayi
Viswanath Nagarajan
Meghana Nasre
Joe Neeman
Ralph Neininger
Jelani Nelson
Huy Nguyen
Ryan O'Donnell
Sewoong Oh
Roberto Oliveira
Yota Otachi
Jiangwei Pan
Konstantinos Panagiotou
Periklis Papakonstantinou
Michal Parnas
Farzad Parvaresh
Chris Peikert
Ron Peled
Tommy Pensyl
Guillem Perarnau
Yury Person
Gábor Pete
Michał Pilipczuk
Yury Polyanskiy
Pawel Pralat
Michal Przykucki
Richard Pymar
Tomasz Radzik
Harry Raecke
Felicia Rassmann
Ran Raz
Alexander Razborov
Daniel Reichman
Omer Reingold
Sebastien Roch
Noga Ron-Zewi
Andrzej Rucinski
Sivan Sabato
Sushant Sachdeva
Rishi Saket
Thomas Sauerwald
Saket Saurabh
Rahul Savani
Roy Schwartz
Rocco Servedio
Ronen Shaltiel
Asaf Shapira
Alexander Sherstov
Igor Shparlinski
Vittoria Silvestri
Arno Siri-Jégousse
Allan Sly
Hao Song
Christian Sommer
Dan Spielman
Joachim Spoerhase
Piyush Srivastava
Alexandre Stauffer
Mike Steele
Daniel Štefankovič
John Steinberger
David Steurer
Benny Sudakov
Nike Sun
Jukka Suomela
Kenjiro Takazawa
Li-Yang Tan
Prasad Tetali
Justin Thaler
Nicolas Trotignon
Madhur Tulsiani
Christopher Umans
Seeun Umboh
Andrew Uzzell
Salil Vadhan

Nithin Mahendra Varma
Juan Vera
Aravindan Vijayaraghavan
Dan Vilenchik
Antonia Wachter-Zeh
Stephan Wagner
Zizhuo Wang
Justin Ward
Lutz Warnke
Thomas Watson
Omri Weinstein
Andreas Wiese
David Woodruff
John Wright
Bang Ye Wu
Patrick Xia
Ning Xie
Chaoping Xing
Guang Yang
Grigory Yaroslavtsev
Yuichi Yoshida
Raphael Yuster
Lenka Zdeborova
Peng Zhang
Yufei Zhao
David Zuckerman

List of Authors

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Fully Dynamic All-Pairs Shortest Paths: Breaking the $O(n)$ Barrier

Ittai Abraham, Shiri Chechik, and Kunal Talwar

Microsoft Research Silicon Valley, Mountain View CA, USA
{ittaia,schechik,kunal}@microsoft.com

Abstract

A fully dynamic approximate distance oracle is a distance reporting data structure that supports dynamic insert edge and delete edge operations. In this paper we break a longstanding barrier in the design of fully dynamic all-pairs approximate distance oracles. All previous results for this model incurred an amortized cost of at least $\Omega(n)$ per operation. We present the first construction that provides constant stretch and $o(m)$ amortized update time. For graphs that are not too dense (where $|E| = O(|V|^{2-\delta})$ for some $\delta > 0$) we break the $O(n)$ barrier and provide the first construction with constant stretch and $o(n)$ amortized cost.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Shortest Paths, Dynamic Algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.1

1 Introduction

A *dynamic distance oracle* (DDO), also known as the *dynamic all pairs shortest path (APSP)*, is a data structure that is capable of efficiently processing an adversarial sequence of delete, insert and distance query operations. A *delete* operation deletes a single edge from the graph. An *insert* operation adds a single edge to the graph. A *query* operation receives a pair of nodes and returns a distance estimation. A *dynamic graph* is some initial graph G and a sequence of delete and insert operations. We say that a dynamic algorithm is only *decremental* if it handles only delete operations, only *incremental* if it handles only insert operations, and *fully dynamic* if it handles both. A dynamic algorithm is only *non-contracting* if it handles both delete and insert but only under the promise that the distances between any two points never get shorter.

A dynamic *approximate distance oracle* has *stretch* k if the returned distance estimate for every pair of nodes is at least the actual distance between them and at most k times their actual distance. A *single-source* dynamic distance oracle (SSDDO) has a fixed source s and all distance queries must involve the source s.

Even for single-source decremental dynamic distance oracles we do not know of any non-trivial bounds on worst-case operation costs. So it is natural to consider amortized costs as the next best measure. The *amortized cost* of a fully dynamic distance oracle is the average cost given a sequence of m operations taken over all possible adversarial sequences and all possible graphs with n vertices that start out with m edges. For a decremental only DDO we can measure the *total cost* as the total cost given an arbitrary sequence of m delete operations taken over all possible graphs with n vertices that start out with m edges. For a non-contracting DDO and a parameter m we define the *total cost* as the total cost given an arbitrary sequence of insert and delete operations such that the initial number of edges plus the total number of insert operations is at most m, taken over all possible graphs with n vertices.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 1–16

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

In this paper we consider fully dynamic DDO for undirected unweighted graphs. For exact distances the best known bound is achieved by Henzinger *et al.* [12] who obtain amortized cost[1] of $O((n^{1.8+o(1)} + m^{1+o(1)})/m)$. When $2+\epsilon$ approximate distances are allowed then Bernstein [5] obtained $\hat{O}(m)$ amortized cost[2]. For larger stretch Baswana, Khurana, and Sarkar [4] obtain, for any k, stretch $4k$ and $\hat{O}(n^{1+1/k})$ amortized update cost.

We note that all previous constructions suffer from an inherent amortized cost of $O(n)$ due to the potential need to run an exact single source shortest path algorithm (e.g. Dijkstra) on the graph G itself or on a sparser subgraph H. Indeed it seems that $O(n)$ amortized cost is a natural barrier for all existing approaches, even when allowing super constant stretch.

The main result of this paper is a new construction that circumvents this barrier and obtains an amortized cost of $o(m)$ and constant stretch. In fact our construction obtains $o(n)$ amortized update and constant stretch for any graph that is not super-dense (formally when $|E| = o(|V|^2)$).

▶ **Theorem 1.** *For any integer k, there exists a fully dynamic DO with amortized expected update time $\tilde{O}(m^{1/2} \cdot n^{1/k})$, query time $O(k^2\rho^2)$, and $2^{O(k\rho)}$ stretch, where $\rho = 1 + \lceil \frac{\log n^{1-1/k}}{\log (m/n^{1-1/k})} \rceil$.*

Note that $\rho \leq k$ and that if $m = n^{1+\epsilon}$ for any constant $\epsilon > 0$ then $\rho = O(1)$. For any graph with $m = n^{2-\delta}$, our algorithm breaks the $O(n)$ barrier with only $O(1)$ stretch.

For any large constant stretch and non-super dense graph our result dominates all previous fully dynamic results. At the extreme, for spare graphs ($|E| = O(|V|)$) and stretch $O(\log n)$ our amortized cost is $o(n^{\frac{1}{2}+\delta})$ for any $\delta > 0$, while [4] require amortized cost $\Omega(n)$.

Our result is obtained by combining two new ingredients. The first is an extremely time-efficient decremental only DDO scheme.

▶ **Theorem 2.** *For any positive integer k, one can maintain a decremental all-pairs shortest paths algorithm for a graph $G = (V, E)$ with stretch $2^{O(\rho k)}$ in total update time $\tilde{O}(mn^{1/k})$ and with $O(k\rho)$ time per query, where $\rho = (1 + \lceil \frac{\log n^{1-1/k}}{\log (m/n^{1-1/k})} \rceil)$.*

This improves on the best known decremental-only dynamic distance oracle of Bernstein and Roddity [6] (that get total update time $\hat{O}(n^{2+\frac{2}{1+stretch}})$) whenever m is $O(n^{2-\delta})$. In particular for sparse graphs with $m = n$, we get total update time $\tilde{O}(n^{1+\frac{1}{k}})$, whereas all previously known results had at least $\Omega(n^2)$ total update time.

The second is a transformation from a decremental only DDO to a fully dynamic DDO that avoids the $\Omega(n)$ worst case insert costs that are present in all previous results. Our reduction is fairly general, and can use other decremental only DDOs. We note that the idea of transforming a decremental algorithm into a fully dynamic algorithm was first suggested by Henzinger and King [11]. In this paper we use this extension in a non-trivial way.

1.1 Related Work

Even and Shiloach, in 1981, presented a decremental SSDDO for undirected, unweighted graphs with $O(n)$ amortized cost and $O(1)$ query time with stretch 1 (exact distances). A similar scheme was independently found by Dinitz [10]. Additional generalizations, optimizations and reductions were studied in [13, 14, 21].

---

[1] As usual, n (respectively, m) is the number of nodes (resp., edges) in the graph.

[2] Throughout, the $\tilde{O}$ notation suppresses polylogarithmic factors and the $\hat{O}$ notation suppresses $n^{O(1/\sqrt{\log n})}$ factors

Ausiello *et al.* [1] presented an incremental DDO for weighted directed graphs with amortized cost $O(n^3 \log n/m)$ and $O(1)$ query time. Henzinger and King showed a decremental DDO for weighted directed graphs with amortized cost $\tilde{O}((n^2/t)+n)$ and $O(t)$ query time.

King [13] presented a fully DDO for unweighted graphs with amortized cost $\tilde{O}(n^{2.5})$ and $O(1)$ query time. Demetrescu and Italiano [9] presented a fully DDO for directed weighted graphs with amortized cost $\tilde{O}(n^{2.5}\sqrt{S})$, where S is the possible number of different weight values in the graph.

Demetrescu and Italiano [8], in a major breakthrough devised a fully dynamic exact DDO for directed general graphs with non negative edge weights, with amortized cost $\tilde{O}(n^2)$. Thorup [15] extended the result to negative edge weights and [16] obtained worst case update time $\tilde{O}(n^{2.75})$.

The dynamic distance oracle problem was also studied when approximated distances are allowed. Much work was on the incremental-only and decremental-only e.g., [2, 3, 19, 6, 20]. Recently, Henzinger *et al.* [12] improved the amortized cost of decremental single source shortest paths (SSSP) for unweighted undirected graphs to $O((n^{1.8+o(1)}+m^{1+o(1)})/m)$.

Fully Dynamic Approximate DDOs for General Graphs: King [13] presented a fully DDO with amortized cost $\tilde{O}(n^2)$, $O(1)$ query time and $(1+\epsilon)$ stretch. Roditty and Zwick [19, 20] presented a fully DDO for any fixed $\epsilon, \delta > 0$ and every $t \leq m^{1/2-\delta}$, with expected amortized cost of $\tilde{O}(mn/t)$ and worst case query time of $O(t)$ and $(1+\epsilon)$ stretch. Note that as $t \leq m^{1/2-\delta}$, the best amortized cost that can be achieved using this algorithm is $\Omega(m^{1/2+\delta}n) > \Omega(m)$. Later, Bernstein [5] presented fully DDO with $O(\log\log\log n)$ query time, $2+\epsilon$ stretch and $\hat{O}(m)$ amortized cost, where $\hat{O}(f(n)) = f(n)n^{O(1/\sqrt{\log n})}$.

Recently, Baswana, Khurana, and Sarkar [4] presented a fully DDO for undirected unweighted graphs breaking the $O(n^2)$ barrier for dense graphs. For an integer parameter k, the construction of [4] has stretch $4k$, $\hat{O}(n^{1+1/k})$ amortized update cost, and $O(\log\log\log n)$ query time.

2 Preliminaries and Notation

Our algorithm is randomized and we assume an oblivious adversary (the sequence of insert and delete operations is determined before the random coins). For simplicity, we describe how to retrieve an estimation on the distances. Our algorithms can also be easily augmented to also report paths. For a graph H, let $V(H)$ be the nodes in H and let $E(H)$ be the edges of H. For an edge $(x,y) \in E(H)$, let $\omega(x,y,H)$ be the weight of the edge (x,y) in the graph H. For a graph H and nodes u and v, $\mathbf{dist}(u,v,H)$ is the distance between u and v in the graph H. Similarly, $\mathbf{dist}(u,S,H)$ for a graph H, node u and a set of nodes S is the minimum distance in H from u to a node in S. For a node v, distance ρ and graph H, let $B(u,\rho,H)$ be the set of nodes at distance at most ρ from u in H. When $H=G$, we sometimes omit it and write simply $\mathbf{dist}(u,v)$ instead of $\mathbf{dist}(u,v,G)$, or $B(u,\rho)$ instead of $B(u,\rho,G)$.

As alluded to earlier, we will often consider distance oracles that work for integer edge-weighted graphs under both edge insertion and deletion, but only under the promise that for every pair of vertices, the distance only increases over time. Thus if we ever insert an edge (u,v), its length will be at least as large as the current shortest path length. We call this the non-contracting dynamic setting. Dynamic distance oracles for this setting will be useful subroutine in our algorithms.

2.1 Existing Decremental SSSP Algorithms

2.1.1 The Decremental SSSP Algorithm of King [13]

Our algorithm uses the decremental SSSP algorithm of King [13] as an ingredient. The properties of King's algorithm are summarized in the following theorem.

▶ **Theorem 3.** *[13] Given a directed graph with positive integer edge weights, a source node s and a distance d, one can decrementally maintains a shortest path tree T from s up to distance d in total time $O(md)$. Moreover, given a node v, one can extract in $O(1)$ time* $\mathbf{dist}(v, s)$ *in case $v \in T$ or determine that $v \notin T$.*

King's algorithm starts by constructing a shortest path tree T rooted at s. Each time an edge $e = (x, y)$ is deleted, where x is in the same connected component as s in $T \setminus e$, an attempt is made to find a substitute edge to y that does not increase the distance from s to y. If such an edge is found then the recovery phase is over. Note that in this case the distances from s to y and to all nodes in y's subtree are unchanged. In case no such edge found, the best edge is chosen, i.e., the edge that connect y on the shortest path possible. The process is continued recursively on all y's children. The crucial property of this algorithm is that it explores the edges of a node v only when the distance from s to v increases. This gives a total running time of $O(md)$ as the distance from s to a node v may increase at most d times before exceeding d.

This analysis of the decremental SSSP algorithm of King [13] works in the decremental only setting but breaks down if we allow edge insertions. Indeed the analysis relies on the fact that the distance from a node to s can change at most d times before exceeding d, which is not true if we allow arbitrary edge insertions. However, if the edges have integer weights and insertions are guaranteed to ensure that the distances do not decrease over time, it is easy to verify that the analysis of King [13] works as it is. Thus Theorem 3 also holds for the non-contracting dynamic setting.

2.1.2 The Decremental Algorithm of Roditty and Zwick [20]

Another ingredient in our algorithm is the approximate decremental APSP algorithm of Roditty and Zwick [20]. Roditty and Zwick [20] showed how to construct a decremental all pairs shortest path data structure DO_{RZ} up to depth d for a given graph H and integer k such that one can answer any distance query in $O(k)$ time within stretch $2k - 1$, and the total update time is $\tilde{O}(mn^{1/k}d)$. More precisely, the dynamic data structure of Roditty and Zwick can be easily tweaked to either return an estimate within $2k - 1$ stretch or determine that $\mathbf{dist}(s, t, H) > d$, and return infinity in this case. In addition, as in the King's algorithm [13], the entire analysis relies on the fact that distances never get shorter and thus also works in the non-contracting setting.

3 Techniques

In this section we outline the high level ideas of our construction for the fully dynamic APSP algorithm. Our algorithm consists of two main parts. The first part is a new decremental APSP algorithm with total update time that can get arbitrarily close to $\tilde{O}(m)$, while paying in the stretch. More precisely, we show for any positive integer k, a decremental APSP algorithm with total update time of $\tilde{O}(mn^{1/k})$ and with stretch $2^{O(k^2)}$ (and $2^{O(k)}$ when $m = n^{1+\epsilon}$). The second part takes the decremental APSP and augments it to accommodate

insertion operations. We provide a general framework for obtaining fully dynamic APSP which can be potentially used with other decremental APSP.

Usually, the hard part in decremental APSP algorithms is in handling long distances. We introduce a new approach that allows efficient handling of all distances. Loosely speaking, we maintain k non-contracting dynamic graphs G_i. The graph G_0 is simply G and we use [20] upto depth $n^{1/k}$ on G_0 to answer distances upto to $n^{1/k}$. We then construct a dynamic non-contracting graph G_1, that dynamically maintains the property that every two nodes whose distance in G is $n^{1/k}$ have a 2-hop path between them in G_1. This implies that distances upto $n^{2/k}$ in G have a path of length $O(n^{1/k})$ in G_1. Hence we use [20] upto depth $n^{1/k}$ on G_1 to answer distances upto to $n^{1/k}$ in G_1 which correspond to distances in G between $n^{1/k}$ to $n^{2/k}$.

In the same way, we iteratively construct a dynamic non-contracting graph G_i, that dynamically maintains the property that every two nodes whose distance in G is $n^{i/k}$ have a 2-hop path between them in G_i. This implies that distances upto $n^{i/k}$ in G have a path of length $O(n^{1/k})$ in G_i. Hence we use [20] upto depth $n^{1/k}$ on G_i to answer distances upto to $n^{1/k}$ in G_i which correspond to distances in G between $n^{i/k}$ to $n^{(i+1)/k}$.

The core difficulty is dynamically maintaining the property that every two nodes whose distance in G is $n^{i/k}$ have a 2-hop path between them in G_i while keeping G_i sparse. For example, consider two nodes that start by having two $n^{i/k}$ long paths between them in G and the 2-hop path maintained in G_i is induced by the first path. Due to edge removals in G of the first path, we now have to discover and maintain in G_i a new 2-hop path that is induced by the second $n^{i/k}$ long path between them in G. Observe that this implies that deleting edges in G may force insert operations of new edges in G_i. In order to control the total cost of this addition we must do two things (1) guarantee that these edges insertions are non-contracting (2) bound the total number of edge insertions.

Suppose we want to maintain distances in G_1 between every two nodes whose distance in G is $\alpha = n^{1/k}$. For simplicity, first assume $k = 2$. Our solution is roughly as follows: we sample $\tilde{O}(\sqrt{n})$ *pivots* A and build and maintain a decremental tree $T(u)$ of radius 3α around each pivot u. We build in G_1 an edge u, v of length 1 between any pivot u and $v \in T(u)$ (in this part edges may be dynamically deleted but none are dynamically inserted). Consider a node w. There are two cases. If $\mathbf{dist}(w, A) \leq 2\alpha$ then there is a nearby pivot that provides the desired 2-hop property for pairs involving w. Otherwise with high probability $|B(w, 2\alpha)| < \sqrt{n}$ and hence for all $x \in B(w, \alpha)$, we have $|B(x, \alpha)| < \sqrt{n}$. So in this case, we *activate* w by building and maintaining a decremental tree $T(w)$ of radius α around w. The main observation is that due to the sparseness condition for activation, for any node y, in the entire decremental sequence on G, y will belong to at most $\sqrt{n}$ trees induced from activated nodes. This implies that we can bound the total number of edges added to G_1 by $\tilde{O}(n^{1+1/2})$. We still have the problem of guaranteeing that these edges insertions are non-contracting in G_1. The solution is to add in G_1 an edge w, v of length 2 between any activated node w and $v \in T(w)$. Just before activating w it must be that (w, v) have a 2-hop path where each hop has length 1 in G_1, and hence choosing length 2 is adequate.

More generally, we maintain k dynamic non-contracting graphs. For each G_ℓ we maintain k subsets $A_1, \ldots, A_k$. The set A_i is of size $\tilde{O}(n^{i/k})$. For each $w \in A_i$, we *activate* w and build a decremental tree of radius $3^{k-i}\alpha$ around w roughly when w is "far enough" from all A_j for all $j < i$. This allows us to bound the total number of edges added to G_ℓ for the set A_i by $\tilde{O}(n^{1+1/k})$. To maintain the non-contracting property we add in G_ℓ an edge w, v of length 2^i between any activated node $w \in A_i$ and $v \in T(w)$. See section 4 for details.

Our approach for the second part is to take our decremental DDO and augment it to

accommodate insertion operations. We provide a general framework for obtaining fully dynamic DO which can be potentially used with other decremental DDOs. At a high level, our approach is to maintain two sub-components: (1) for delete operations we maintain a decremental DDO (2) for insert operations we use a *sketch-graph* data structure that maintains an approximate distance oracle over all the newly inserted edges. Once sub-component (2) becomes too large we re-build the two components from scratch and hence obtaining good amortized guarantees. The key advantage is that our costs are proportional to the number of newly inserted edges, not the total number of edges in the graph. During a query we use both components and combine their results to find a low stretch estimation. We also need to update the sketch-graph appropriately during each delete operation. In order to get a smaller update time we exploit some additional properties in the construction of Roditty and Zwick [20].

4 New Decremental Shortest Paths

4.1 The Main Building Block

4.1.1 Properties

In this section we present an algorithm that takes as input three integers k, α, and m_r and a dynamic graph G_r where the graph G_r is guaranteed to satisfy the following properties.

(a) All edge weights in G_r are in $\{1, 2, \ldots, 2^{k-1}\}$ (both the initial edges and the edges that may be dynamically added).

(b) The graph G_r has the property that distances never decrease over time (but it may happen that edges are both added and removed over time).

(c) The initial number of edges in G_r plus the total number of edge insertion operation on G_r is at most m_r.

The algorithm outputs a non-contracting dynamic data structure that produces a dynamic graph G_{r+1} with the following properties:

1. The edge weights in G_{r+1} are in $\{1, 2, \ldots, 2^{k-1}\}$ (both the initial edges and the edges that may be dynamically added).
2. Every two nodes at distance at most α in G_r have a two hop path between them in G_{r+1} of length at most 2^{k-1}.
3. At any point, the non-contracting data structure maintains:

$$\frac{\mathbf{dist}(u, v, G_r)}{(3^{k-1}\alpha)} \leq \mathbf{dist}(u, v, G_{r+1}) \leq 2^{k-1} \cdot \left(\frac{\mathbf{dist}(s, t, G_r)}{(\alpha - 2^{k-1})} + 1 \right)$$

4. The dynamic non-contracting data structure for G_{r+1} incurs a total cost of $\tilde{O}(k3^k n^{1/k} \alpha m_r)$.
5. The non-contracting data structure maintains that the graph G_{r+1} has the property that distances never decrease over time (but it may happen that edges are both added and removed over time).
6. The initial number of edges in G_{r+1} plus the total number of edge insertion operation on G_{r+1} is at most $\tilde{O}(kn^{1+1/k})$.

4.1.2 Constructing the Decremental Distance Oracle

We define a sequence of sets $A_1, \ldots, A_k$ as follows: The set A_k is simply V. For $1 < i \leq k-1$ set A_i is a sample of V independently at random with probability $c \log n / n^{1-i/k}$ (for some small constant c). Thus the set A_i contains in expectation $\tilde{O}(n^{i/k})$ nodes.

For each $v \in A_i$ we will define a *condition* under which we *activate* v and hereafter build and maintain a decremental SSSP tree $T(v)$ for depth $3^{k-i}\alpha$ on graph G_r. Loosely speaking, for every node v in A_i for every $i > 1$, the algorithm constructs a decremental SSSP tree $T(v)$ once all the distances $\mathbf{dist}(v, A_j, G_r)$ for every $1 \leq j < i$ are "sufficiently" large.

We say that a node $v \in A_i$ is *i-active* if the tree $T(v)$ was already constructed, otherwise it is *i-inactive*. All nodes in A_1 are initially active, namely, for every node v in A_1 maintain a decremental SSSP tree $T(v)$ up to depth $3^{k-1}\alpha$ on the graph G_r. For every index $1 \leq i \leq k-1$ and a node u, the algorithm maintains in a heap H_i^u the distances $\mathbf{dist}(v, u, G_r)$ for every $v \in A_i$ such that v is i-active and $u \in T(v)$. Let $H_i^u.min$ be the minimum in the heap. For every $i > 1$ and i-inactive node $v \in A_i$: if the condition $H_j^u.min > (3^{k-j} - 3^{k-i})\alpha$ for every $j < i$ holds, then the algorithm constructs and maintains a decremental SSSP tree $T(v)$ up to depth $3^{k-i}\alpha$ on the graph G_r. Each time the distance $\mathbf{dist}(u, v, T(v))$ for some $u \in T(v)$ changes, the algorithm updates the relevant heap and checks if u should be activated.

The graph G_{r+1} is constructed and maintained as follows. For every node $v \in A_i$ and $u \in T(v)$, add an edge between u and v of weight 2^{i-1}. Once a node u is removed from $T(v)$, remove the corresponding edge from G_{r+1}. Similarly, once v is i-activated and $T(v)$ for $v \in A_i$ is constructed add all edges (u, v) for every $u \in T(v)$ with weight 2^{i-1}. In each change of G_r, the algorithm first updates all the trees $T(v)$, then adds the relevant edges to G_{r+1}, and then removes the relevant edges from G_{r+1}. Adding the edges before the deletions ensures that distances are never decrease in G_{r+1}. This concludes the construction.

Observe that one change in G_r may lead multiple changes in G_{r+1} (a removal of an edge may increase some distances in trees $T(v)$ that may lead to the construction of new trees).

4.1.3 Analysis

The next claim bounds the number of trees $T(v)$ a node u may belong to in the entire run of the algorithm.

▶ Claim 4. W.h.p. every node u belongs to at most $\tilde{O}(k \cdot n^{1/k})$ trees $T(v)$ in the entire run of the algorithm.

Proof. There are $\tilde{O}(n^{1/k})$ nodes in A_1 in expectation, therefore u may belong to $\tilde{O}(n^{1/k})$ trees $T(v)$ such that $v \in A_1$.

We claim that u belongs to a tree $T(v)$ for $v \in A_i$ and $i > 1$, only if $\mathbf{dist}(u, A_{i-1}, G_r) > 3^{k-i}\alpha$. To see this, assume that $\mathbf{dist}(u, A_{i-1}, G_r) \leq 3^{k-i}\alpha$. Assume, towards contradiction, that $u \in T(v)$ for some $v \in A_i$. Recall that the depth of $T(v)$ is $3^{k-i}\alpha$. We get that $\mathbf{dist}(v, A_{i-1}, G_r) \leq \mathbf{dist}(v, u, G_r) + \mathbf{dist}(u, A_{i-1}, G_r) \leq (3^{k-i} + 3^{k-i})\alpha \leq 2 \cdot 3^{k-i}\alpha$. Hence there is a node $w \in A_{i-1}$ such that $\mathbf{dist}(v, w, G_r) \leq 2 \cdot 3^{k-i}\alpha$.

We need to consider two cases. The first case is when w is $(i-1)$-active and the second case is when w is $(i-1)$-inactive. Consider the case where w is $(i-1)$-active. The tree $T(w)$ is constructed up to depth $3^{k-(i-1)}\alpha = 3^{k-i+1}\alpha$. As $\mathbf{dist}(v, w, G_r) \leq (2 \cdot 3^{k-i})\alpha < 3^{k-i+1}\alpha$ we have $v \in T(w)$. Note that in this case,

$$\begin{aligned} H_{i-1}^v.min &\leq \mathbf{dist}(v, w, G_r) \\ &\leq 2 \cdot 3^{k-i}\alpha \\ &= (3^{k-i+1} - 3^{k-i})\alpha \\ &= (3^{k-(i-1)} - 3^{k-i})\alpha. \end{aligned}$$

Hence, by definition, the tree $T(v)$ was not supposed to be constructed yet, which is a contradiction.

Consider now the second case, where w is $(i-1)$-inactive. The node w is $(i-1)$-inactive only if there is a j-active node $z \in A_j$ for some $j < i-1$ such that $w \in T(z)$ and $\mathbf{dist}(w, z, G_r) \leq (3^{k-j} - 3^{k-(i-1)})\alpha$.

It follows that

$$\begin{aligned}
\mathbf{dist}(v, z, G_r) &\leq \mathbf{dist}(v, w, G_r) + \mathbf{dist}(w, z, G_r) \\
&\leq (2 \cdot 3^{k-i})\alpha + (3^{k-j} - 3^{k-(i-1)})\alpha \\
&= (2 \cdot 3^{k-i} + 3^{k-j} - 3^{k-i+1})\alpha \\
&= (2 \cdot 3^{k-i} + 3^{k-j} - 3 \cdot 3^{k-i})\alpha \\
&= (3^{k-j} - 3^{k-i})\alpha.
\end{aligned}$$

Note that $v \in T(z)$. It follows that, $H_j^v.min \leq \mathbf{dist}(v, z, G_r) \leq (3^{k-j} - 3^{k-i})\alpha$. Hence, by definition, the tree $T(v)$ was not supposed to be constructed yet, which is a contradiction.

It follows that, $\mathbf{dist}(u, A_{i-1}, G_r) > 3^{k-i}\alpha$. Hence, by applying Chernoff's bound [3] we get that w.h.p. $|B(u, 3^{k-i}\alpha)| \leq n^{1-(i-1)/k}$. The set A_i contains every node independently at random with probability $c \log n / n^{1-i/k}$. Hence in expectation we have $|B(u, 3^{k-i}\alpha) \cap A_i| \leq c \log n / n^{1-i/k} \cdot n^{1-(i-1)/k} = \tilde{O}(n^{1/k})$, as required. ◀

We next show that G_{r+1} satisfies the desired properties.

▶ **Lemma 5.** *Suppose that the input dynamic graph G_r satisfies properties (a)-(c). Then the graph G_{r+1} satisfies properties 1–6.*

Proof. It is not hard to verify property (1), that all edges in G_{r+1} are of weight in $\{1, 2, \ldots, 2^{k-1}\}$.

We now prove property (2), that is, for every two nodes u and v such that $\mathbf{dist}(u, v, G_r) \leq \alpha$, there is a path between u and v in G_{r+1} of at most two hop and of length at most 2^{k-1}. To see this, recall that $A_k = V$. If v is k-active then the tree $T(v)$ is constructed up to depth α and as $\mathbf{dist}(u, v, G_r) \leq \alpha$ we have $u \in T(v)$. Therefore, by construction the graph G_{r+1} contains the edge (u, v) of weight 2^{k-1}, as required. So assume now v is k-inactive. By construction if v is k-inactive then there is an index $j < k$ and a j-active node $w \in A_j$ such that $v \in T(w)$ and $\mathbf{dist}(v, w, G_r) \leq (3^{k-j} - 1)\alpha$. Note that $\mathbf{dist}(u, w, G_r) \leq \mathbf{dist}(u, v, G_r) + \mathbf{dist}(v, w, G_r) \leq \alpha + (3^{k-j} - 1)\alpha \leq 3^{k-j}\alpha$. As the tree $T(w)$ contains all nodes at distance in G_r at most $3^{k-j}\alpha$ from w, we get that $u \in T(w)$. By construction, the graph G_{r+1} contains both edges (w, v) and (w, u), both of weight 2^{j-1}. Hence G_{r+1} has a two hop path between u and v of length $2^j \leq 2^{k-1}$, as required.

To see property (3) consider two nodes u and v. We need to show $\mathbf{dist}(u, v, G_r)/(3^{k-1}\alpha) \leq \mathbf{dist}(u, v, G_{r+1}) \leq 2^{k-1}(\mathbf{dist}(s, t, G_r)/\alpha + 1)$.

Let us first show the first inequality, that is, $\mathbf{dist}(u, v, G_r)/(3^{k-1}\alpha) \leq \mathbf{dist}(u, v, G_{r+1})$. Let $P(u, v, G_{r+1})$ be the shortest path between u and v in G_{r+1}. Let (x, y) be an edge on $P(u, v, G_{r+1})$. Recall that by construction as (x, y) is an edge in G_{r+1} then there is an index i such that $x \in A_i$, and $y \in T(x)$, or $y \in A_i$, and $x \in T(y)$. Assume w.l.o.g. that $x \in A_i$ and $y \in T(x)$.

[3] Note that the claim needs to hold w.h.p. for every considered graph during the entire running of the algorithm, note however that as there are at most $m_r \leq n^2$ deletions and therefore at most n^2 considered graphs. Hence by setting the constant c to be large enough we can show that the claim holds w.h.p. for every considered graph.

Note that $\mathbf{dist}(x,y,G_{r+1}) = 2^{i-1}$ and that $\mathbf{dist}(x,y,G_r) \leq 3^{k-i}\alpha$. We get

$$\begin{aligned}\mathbf{dist}(x,y,G_r) &\leq 3^{k-i}\alpha \\ &= 3^{k-i}\alpha/2^{i-1}\cdot 2^{i-1} \\ &= 3^{k-i}\alpha/2^{i-1}\cdot \mathbf{dist}(x,y,G_{r+1}) \\ &\leq 3^{k-1}\alpha\cdot \mathbf{dist}(x,y,G_{r+1}).\end{aligned}$$

Hence, $\mathbf{dist}(x,y,G_r)/(3^{k-1}\alpha) \leq \mathbf{dist}(x,y,G_{r+1})$. It follows that

$$\begin{aligned}\mathbf{dist}(u,v,G_{r+1}) &= \sum_{(x,y)\in P(u,v,G_{r+1})} \omega(x,y,G_{r+1}) \\ &\geq \sum_{(x,y)\in P(u,v,G_{r+1})} \mathbf{dist}(x,y,G_r)/(3^{k-1}\alpha) \\ &\geq \mathbf{dist}(u,v,G_r)/(3^{k-1}\alpha),\end{aligned}$$

as required.

We now turn to prove the second inequality, namely, $\mathbf{dist}(u,v,G_{r+1}) \leq 2^{k-1}(\mathbf{dist}(u,v,G_r)/(\alpha-2^{k-1})+1)$. Consider the shortest path $P(u,v,G_r)$ between u and v in G_r. Let $x_0 = u$ and let x_i be the furthest away node on $P(u,v,G_r)$ from x_{i-1} such that $\mathbf{dist}(u,v,G_r) \leq \alpha$. As the maximum edge weight in G_r is 2^{k-1} it is not hard to verify that $\mathbf{dist}(x_{i-1},x_i,G_r) < \alpha - 2^{k-1}$. In addition, by property (2) we have that x_{i-1} and x_i are connected by a two hop path of length at most 2^{k-1}.

Hence, u and v are connected by a path in G_{r+1} of length at most $\lceil \mathbf{dist}(u,v,G_r)/(\alpha - 2^{k-1})\rceil \cdot 2^{k-1}$. Therefore,

$$\begin{aligned}\mathbf{dist}(u,v,G_{r+1}) &\leq \lceil \mathbf{dist}(u,v,G_r)/(\alpha-2^{k-1})\rceil \cdot 2^{k-1} \\ &\leq (\mathbf{dist}(u,v,G_r)/(\alpha-2^{k-1})+1)\cdot 2^{k-1}.\end{aligned}$$

We get that, $\mathbf{dist}(u,v,G_r)/(3^{k-1}\alpha) \leq \mathbf{dist}(u,v,G_{r+1}) \leq (\mathbf{dist}(s,t,G_r)/(\alpha-2^{k-1})+1)\cdot 2^{k-1}$.

Let us turn to prove property (4). By claim 4 all nodes belong to $\tilde{O}(kn^{1/k})$ trees. Each tree is maintained up to depth $3^{k-1}\alpha$ (or less). Let $\deg_r(u)$ be the degree of u in G_r. Consider a tree $T(v)$, the total time for maintaining $T(v)$ is $\sum_{u\in T(v)} 3^{k-1}\alpha \deg_r(u)$. Thus for all trees $\sum_{v\in V, u\in T(v)} 3^{k-1}\alpha\deg_r(u) = \tilde{O}(k3^{k-1}n^{1/k}\alpha m_r)$.

We now turn to property (5), that is, edges may be added to G_{r+1} but distances are only increasing. We need to show that when we add an edge (u,v) with weight $\omega(u,v)$ then $\mathbf{dist}(u,v,G_{r+1}) \leq \omega(u,v)$. Recall that an edge (u,v) is added to G_{r+1} when the tree $T(v)$ for some $v \in A_i$ is constructed. The tree $T(v)$ is constructed once the distances $H^u_j.min > (3^{k-j}-3^{k-i})\alpha$ for every $j < i$. Namely, just before this change we had $H^u_j.min \leq (3^{k-j}-3^{k-i})\alpha$ for some $j < i$. It follows that there exists a node $w \in A_j$ such that $\mathbf{dist}(v,w,G_r) = (3^{k-j}-3^{k-i})\alpha$. Note that $\mathbf{dist}(v,u,G_r) \leq 3^{k-i}\alpha$. Hence $\mathbf{dist}(u,w,G_r) \leq \mathbf{dist}(u,v,G_r) + \mathbf{dist}(v,w,G_r) \leq 3^{k-i}\alpha + (3^{k-j}-3^{k-i})\alpha = 3^{k-j}\alpha$. It follows that $u \in T(w)$. We get that the edges (u,w) and (w,v) existed in G_{r+1} just before this change. In addition, $\omega(u,w) = 2^{j-1} \leq 2^{i-1}$ and $\omega(w,v) = 2^{j-1} \leq 2^{i-1}$. The algorithm adds an edge (u,v) with $\omega(u,v) = 2^i$. It is not hard to see that the distance (u,v) did not decrease by adding the edge (u,v).

Property (6), i.e., that the total number of edges in G_{r+1} is $\tilde{O}(kn^{1+1/k})$ easily follows from the fact that every node belong to $\tilde{O}(kn^{1/k})$ trees. ◀

4.2 The Main Decremental Construction

In this section we show our decremental algorithm.

Construction. Let us start with describing the construction. Set $G_0 = G$. Construct G_1 using the data structure from the previous section on G_0 with $\alpha_1 = n^{1/k}$ and k. For every $1 < i \leq \rho$, construct and maintain G_i using the data structure from the previous section on G_{i-1} with $\alpha_i = 2^k\lceil \frac{m}{n^{1-1/k}} \rceil$ and k. (The reason for considering different α's for the different levels if to achieve a better stretch for not super sparse graphs). For every $0 \leq i \leq \rho$ construct the data structure DO^i_{RZ} of Roditty and Zwick [20] up to distances α_i on the graph G_i. The concludes the construction.

Answering a Distance Query. The algorithm for answering distance query given two nodes s and t is as follows. Find the first i such that DO^i_{RZ} returns an estimation on $\mathbf{dist}(s,t)$, i.e., the first i such that s and t are at distance at most $2\alpha_i$ from one another in G_i. Let $\mu_0 = 1$ and $\mu_i = \alpha_1 \cdot \alpha_2^{i-1} \cdot 3^{i(k-1)}$ for $i > 0$. Return $\mu_i DO^i_{RZ}(s,t)$.

It is not hard to see that the query time is $O(k \cdot \rho)$ as the algorithm invokes the Roditty and Zwick [20] query algorithm at most ρ times, where each distance query of Roditty and Zwick [20] takes $O(k)$ time.

Analysis. The next auxiliary claim shows that every two nodes whose distance in G is at most $\alpha_1(\alpha_2/2^k)^{i-1}$ are connected by a path of at most two hop in G_i.

▶ Claim 6. Every two nodes u and v such that $\mathbf{dist}(u,v,G) \leq \alpha_1(\alpha_2/2^k)^{i-1}$ are connected in G_i by a path of at most 2-hop.

Proof. We prove it by induction on i. For $i = 1$, the proof follows by property (2) on G_1. Assume correctness for every j such that $1 \leq j < i$ and consider i. Consider two nodes u and v such that $\mathbf{dist}(u,v,G) \leq \alpha_1(\alpha_2/2^k)^{i-1}$.

Consider the shortest path $P(u,v)$ between u and v in G. Let $u = x_0$ and $x_r = v$ for $r = \lceil \mathbf{dist}(u,v,G)/(\alpha_1(\alpha_2/2^k)^{i-2}) \rceil$. For $1 \leq \ell < r$, let x_ℓ be the next node on the path $P(u,v)$ (closer to v) such that $\mathbf{dist}(x_{\ell-1}, x_\ell) = \alpha_1(\alpha_2/2^k)^{i-2}$.

By the induction hypothesis x_ℓ and $x_{\ell+1}$ are connected by a two-hop path in G_{i-1}. As the weights in G_{i-1} are at most 2^{k-1}. We get that the length of the path between u and v in G_{i-1} is at most

$$\begin{aligned}
\mathbf{dist}(u,v,G_{i-1}) &\leq 2^k\lceil \mathbf{dist}(u,v,G)/(\alpha_1(\alpha_2/2^k)^{i-2}) \rceil \\
&\leq 2^k\lceil \alpha_1(\alpha_2/2^k)^{i-1}/(\alpha_1(\alpha_2/2^k)^{i-2}) \rceil \\
&\leq 2^k\lceil 2^k m/n^{1-1/k}/2^k \rceil \\
&= 2^k m/n^{1-1/k} \\
&= \alpha_2
\end{aligned}$$

By property (2) we have that u and v are connected by a two-hop path in G_i. ◀

▶ Claim 7. Let i be the minimal index such that DO^i_{RZ} returns an estimation on $\mathbf{dist}(s,t)$. If $i > 0$, then $\mathbf{dist}(s,t,G) \geq \alpha_1(\alpha_2/2^k)^{i-1}$.

Proof. It is not hard to see by Claim 6 that if $\mathbf{dist}(s,t,G) \leq \alpha_1(\alpha_2/2^k)^{i-1}$ then there is $2\alpha_2/2^k$ hop paths from s to t in G_{i-1}, namely, $\mathbf{dist}(s,t,G_{i-1}) \leq \alpha_2$. Therefore by construction DO^{i-1}_{RZ} is supposed to return an estimate on $\mathbf{dist}(s,t)$. It follows that $\mathbf{dist}(s,t,G) \geq \alpha_1(\alpha_2/2^k)^{i-1}$. ◀

The next lemma bounds the stretch of the algorithm.

▶ **Lemma 8.** *The distance* $\hat{\mathbf{dist}}(s,t)$ *returned by the algorithm satisfies* $\mathbf{dist}(s,t) \leq \hat{\mathbf{dist}}(s,t) \leq k(2k-1)6^{\rho k}\mathbf{dist}(s,t)$.

Proof. Let i be the minimal index such that DO^i_{RZ} returns an estimation on $\mathbf{dist}(s,t)$. If $i = 0$, then $\hat{\mathbf{dist}}(s,t) = DO^0_{RZ}(s,t)$. Note that $\mathbf{dist}(s,t,G) \leq DO^0_{RZ}(s,t) \leq (2k-1)\mathbf{dist}(s,t,G)$ and we are done. So assume $i > 0$.

Let us first prove the first direction, that is, $\mathbf{dist}(s,t) \leq \hat{\mathbf{dist}}(s,t)$. By applying property (3) recursively we have $\mathbf{dist}(s,t,G_i) \geq \mathbf{dist}(s,t,G_0)/(3^{i\cdot(k-1)}\alpha_1\alpha_2^{i-1})$. We get,

$$\begin{aligned}
\hat{\mathbf{dist}}(s,t) &= \alpha_1 \cdot \alpha_2^{i-1} \cdot 3^{i(k-1)} \cdot DO^i_{RZ}(s,t) \\
&\geq \alpha_1 \cdot \alpha_2^{i-1} \cdot 3^{i(k-1)} \cdot \mathbf{dist}(s,t,G_i) \\
&\geq \alpha_1 \cdot \alpha_2^{i-1} \cdot 3^{i(k-1)} \cdot \mathbf{dist}(u,v,G_0)/(3^{i\cdot(k-1)}\alpha_1\alpha_2^{i-1}) \\
&= \mathbf{dist}(u,v,G_0).
\end{aligned}$$

We are left to show the other direction, that is, $\hat{\mathbf{dist}}(s,t) \leq k(2k-1)6^{\rho k}\mathbf{dist}(s,t,G_0)$.

Recall that $\alpha_1, \alpha_2 > 2^k$. By property (3) and straightforward calculations we have the following.

$$\begin{aligned}
\mathbf{dist}(s,t,G_i) &\leq 2^{k-1}(\mathbf{dist}(s,t,G_{i-1})/(\alpha_2 - 2^{k-1}) + 1) \\
&= 2^{k-1}\mathbf{dist}(s,t,G_{i-1})/(\alpha_2 - 2^{k-1}) + 2^{k-1} \\
&\leq 2^{i(k-1)}\mathbf{dist}(s,t,G_0)/((\alpha_2 - 2^{k-1})^{i-1} \cdot (\alpha_1 - 2^{k-1})) + i \cdot 2^{k-1} \\
&\leq 2^{i(k-1)}\mathbf{dist}(s,t,G_0)/((\alpha_2/2)^{i-1} \cdot (\alpha_1/2)) + i \cdot 2^{k-1} \\
&\leq 2^{ik}\mathbf{dist}(s,t,G_0)/((\alpha_2)^{i-1} \cdot (\alpha_1)) + i \cdot 2^{k-1}.
\end{aligned}$$

By Claim 7 we have $\mathbf{dist}(s,t,G) \geq \alpha_1(\alpha_2/2^k)^{i-1}$. Hence,

$$\begin{aligned}
\hat{\mathbf{dist}}(s,t) &= \alpha_1 \cdot \alpha_2^{i-1} \cdot 3^{i(k-1)} \cdot DO^i_{RZ}(s,t) \\
&\leq \alpha_1 \cdot \alpha_2^{i-1} \cdot 3^{i(k-1)} \cdot (2k-1)\mathbf{dist}(s,t,G_i) \\
&\leq \alpha_1 \cdot \alpha_2^{i-1} \cdot 3^{i(k-1)} \cdot (2k-1)(2^{ik}\mathbf{dist}(s,t,G_0)/((\alpha_2)^{i-1} \cdot (\alpha_1)) + i \cdot 2^{k-1}) \\
&\leq (2k-1)6^{ik}\mathbf{dist}(s,t,G_0) + i\alpha_1 \cdot \alpha_2^{i-1} \cdot 3^{i(k-1)} \cdot (2k-1) \cdot 2^{k-1} \\
&\leq (2k-1)6^{ik}\mathbf{dist}(s,t,G_0) + i \cdot 3^{i(k-1)} \cdot (2k-1) \cdot 2^{k-1} \cdot \mathbf{dist}(s,t,G_0) \\
&\leq k(2k-1)6^{ik}\mathbf{dist}(s,t,G_0).
\end{aligned}$$

as required. ◀

We conclude the following.

▶ **Theorem 9.** *For any positive integer* k*, one can maintain a decremental all-pairs shortest paths algorithm for a given graph* $G = (V,E)$ *with stretch* $2^{O(\rho k)}$ *in total update time* $\tilde{O}(mn^{1/k})$ *and with* $O(k\rho)$ *time per query, where* $\rho = (1 + \lceil \frac{\log n^{1-1/k}}{\log(m/n^{1-1/k})} \rceil)$

5 Fully Dynamic Approximate All-Pairs Shortest Paths

Our fully dynamic data structure will maintain the decremental data structure from the last section, along with an auxiliary data structure to "remember" the insertions. To prevent this second data structure from becoming too large, we will periodically rebuilid the complete data structure (i.e. both the decremental part and the auxiliary part). Throughout, $G = (V, E)$ is the updated graph, namely, the graph after all the updates that have occurred so far. Let D be the set of deletions and U be the set of insertions since the data structure was last rebuilt. Let $\hat{G}$ be the graph when the data structure was last rebuilt. Let G_D be the graph obtained by deleting the set of edges D from $\hat{G}$.

The main idea of our fully dynamic algorithm is the following: We maintain two data structures, the first is a decremental distance oracle **Dec** and the second is an *auxiliary distance oracle* M whose goal is to handle edge insertions. More specifically, the decremental distance oracle is capable of answering approximate distance queries in the graph G_D, and the sketch distance oracle is capable of answering approximate distance queries in the graph G, but only between nodes in $V(U)$, where $V(U)$ is the set of all nodes incident to some edge in U.

In addition, for every node $v \in V$, we ensure a simple access to some "close" nodes in $V(U)$, called the *pivots* of v. To answer distance queries between a pair of nodes s and t, the algorithm computes for every two pivots $p(s)$ of s and $p(t)$ of t the distance $\mathbf{Dec}(s, p(s)) + M(p(s), p(t)) + \mathbf{Dec}(p(t), t)$ and returns the minimum over all pairs of pivots, where $\mathbf{Dec}(u, v)$ is the distance returned by invoking the query algorithm of **Dec** on (u, v). Let $P(s, t, G)$ be the shortest path from s to t in G. If the path $P(s, t, G)$ does not contain edges that were inserted since the data structure was last constructed then $\mathbf{Dec}(s, t)$ gives a good estimation on $\mathbf{dist}(s, t, G)$. Otherwise, we show that there must be nodes $p(s), p(t) \in V(U)$ such that $\mathbf{Dec}(s, p(s)) + M(p(s), p(t)) + \mathbf{Dec}(p(t), t)$ is a good estimation on $\mathbf{dist}(s, t, G)$.

When the sketch distance oracle becomes too "large" we simply construct the data structure from scratch.

An important advantage of our scheme is that it is quite general. One can plug in it any decremental and sketch distance oracles.

5.1 Fully Dynamic APSP Beyond $O(n)$

We show a fully dynamic APSP algorithm with amortized update time $\tilde{O}(m^{1/2} \cdot n^{1/k})$ with $2^{O(k\rho)}$ stretch. Note that $\rho \leq k$ and that if $m = n^{1+\epsilon}$ for constant ϵ then $\rho = O(1)$. For any graph with $m = n^{2-\delta}$ for some constant $\delta > 0$, our algorithm can break the $O(n)$ barrier with only $O(1)$ stretch.

We use the decremental distance oracle **Dec** from Section 4 to get our fully dynamic APSP. In order to get a fast query time we do not treat **Dec** as a black box, but rather exploit some additional properties in the construction of Roditty and Zwick [19, 20]. As mentioned earlier in our decremental distance oracle **Dec** on each graph G_i our algorithm invokes the Roditty and Zwick [19, 20] decremental algorithm. Roditty and Zwick showed how to maintain the Thorup-Zwick distance oracle decrementally for distances smaller than $\overline{d}$ with total update time $O(\overline{d}mn^{1/k})$. Let us first outline the construction of Thorup-Zwick distance oracle [18]. We will then see how to exploit some properties in the Thorup-Zwick distance oracle [18] in order to get our efficient fully dynamic algorithm.

5.1.1 The Static Distance Oracle of Thorup and Zwick

We outline the construction of the Thorup-Zwick distance oracle [18]. For a given positive integer k, construct the sets $V = A_0 \supseteq A_1 \supseteq \cdots \supseteq A_{k-1}$ as follows: The i-th level A_i is constructed by sampling the vertices of A_{i-1} independently at random with probability $n^{-1/k}$ for $1 \leq i \leq k-1$. The set A_k is set to be the empty set. Next, for every vertex v, define the bunch of v as follows $B(v) = \bigcup_{i=0}^{k-1} B_i(v)$, where $B_i(v) = \{u \in A_i \setminus A_{i+1} \mid \mathbf{dist}(v,u) < \mathbf{dist}(v, A_{i+1})\} \cup \{p_i(v)\}$. The pivot $p_i(v)$ is the closest vertex to v in A_i (break ties arbitrarily).

The Thorup-Zwick Data Structure: For every vertex v store $B(v)$ and for every vertex $w \in B(v)$ store $\mathbf{dist}(w,v)$. In addition, for every vertex v and every index i where $1 \leq i \leq k-1$ store $p_i(v)$ and $\mathbf{dist}(v, p_i(v))$.

The Thorup-Zwick Query Algorithm: Let i be the first index such that either $p_i(s) \in B(t)$ or $p_i(t) \in B(s)$. Assume w.l.o.g. that $p_i(s) \in B(t)$. Return $\mathbf{dist}(s, p_i(s)) + \mathbf{dist}(p_i(s), t)$.

It was shown in [18] that for every $j \leq i$, $\mathbf{dist}(s, p_j(s)) \leq (j-1)\mathbf{dist}(s,t)$ and $\mathbf{dist}(t, p_j(s)) \leq j\mathbf{dist}(s,t)$. Similarly for every $j \leq i$, $\mathbf{dist}(t, p_j(t)) \leq (j-1)\mathbf{dist}(s,t)$ and $\mathbf{dist}(s, p_j(t)) \leq j\mathbf{dist}(s,t)$. Combining it with the fact that $i \leq k-1$, we get the $2k-1$ stretch.

In our case, in order to get fast query we will sometime have only access to the pivots of s. It was shown in [17] that even if we check only the pivots of s and take the first i such that $p_i(s) \in B(t)$ then $\mathbf{dist}(s, p_i(s)) + \mathbf{dist}(p_i(s), t) \leq (4k-3)\mathbf{dist}(s,t)$.

In addition, Thorup and Zwick showed that the expected size of the bunch is $O(k \cdot n^{1/k})$. Finally, Thorup and Zwick showed that in the static case constructing their data structure can be done in time $\tilde{O}(m \cdot n^{1/k})$ time.

5.1.2 Our Fully Dynamic All-Pairs Shortest Paths

As mentioned earlier, we use the decremental distance oracle **Dec** from Section 4. In our decremental distance oracle **Dec** on each graph G_i our algorithm invokes the Roditty and Zwick [19, 20] decremental distance oracle. For each i and a node v, let $B^i(v)$ be the bunch of v in DO^i_{RZ} and let $p^i_j(v)$ be the j'th pivot of v in DO^i_{RZ}. Recall that our decremental algorithm **Dec** returns $\mu_i \cdot DO^i_{RZ}(s,t)$ for the first i such that DO^i_{RZ} returns an estimation on $\mathbf{dist}(s,t)$. In addition, we have that either $DO^i_{RZ}(s,t) = DO^i_{RZ}(s, p^i_{i_1}(s)) + DO^i_{RZ}(p^i_{i_1}(s), t)$ for some $1 \leq i_1 \leq k-1$ or $DO^i_{RZ}(s,t) = DO^i_{RZ}(s, p^i_{i_2}(t)) + DO^i_{RZ}(p^i_{i_2}(t), t)$ for some $1 \leq i_2 \leq k-1$. Moreover, for nodes x, y such that $x \in B^i(y)$ we have $DO^i_{RZ}(x,y) = \mathbf{dist}(x, y, G_i)$.

After $m^{1/2}$ insertions we reconstruct the data structure from scratch. Our data structure contains two main parts **Dec** and an auxiliary distance oracle M. In addition, we maintain the set U containing all the edges that were added since the last time the data structure was constructed.

(Re)Construction: Construct **Dec** as described in Section 4 and an empty set U.

Deletion Operation for an Edge e: We do the following two steps.

1. Invoke the deletion operation of **Dec** for the edge e, and
2. Renew the data structure M as follows: Construct a graph H, initially set to be empty. Add the set of nodes $V(U)$ to H. For every node $v \in V(U)$ and $1 \leq i \leq \rho$, add $B^i(v)$ to H. Add edges from v to every node x in $B^i(v)$, set the weight of the edge to be $\mu_i DO^i_{RZ}(x,v) = \mu_i \mathbf{dist}(x, v, G_i)$. In addition add the set of edges U to H. Compute the Thorup-Zwick distance oracle $DO_{TZ,H}$ on H with parameter k. Store $DO_{TZ,H}$.

Insertion Operation for an Edge $\mathbf{e}=(\mathbf{x},\mathbf{y})$: Add the edge to U. If $|U| \geq m^{1/2}$ then reconstruct the data structure. Otherwise, renew the data structure M (as explained in the deletion operation).

Query Operation Given Two Nodes s and t: Let $DO_{TZ,H}(x,y)$ be the distance estimate returned by the Thprup-Zwick query algorithm on x and y, if either $x \notin V(H)$ or $y \notin V(H)$ then we just set $DO_{TZ,H}(x,y)=\infty$.

Return the minimum distance between $\mathbf{Dec}(s,t)$ and the minimum

$$\min\{\, \mu_{i_1} DO^{i_1}_{RZ}(s,p^{i_1}_{j_1}(s)) + DO_{TZ,H}(p^{i_1}_{j_1}(s),p^{i_2}_{j_2}(s)) + \mu_{i_2} DO^{i_2}_{RZ}(t,p^{i_2}_{j_2}(t)) \mid 1 \leq j_1,j_2 \leq k-1, 1 \leq i_1,i_2 \leq \rho\}.$$

Analysis: We now turn to the analysis. The key lemma that we present next, bounds the stretch of our dynamic distance oracle.

▶ **Lemma 10.** *Consider two nodes s and t, the distance $\hat{\mathbf{dist}}(s,t)$ returned by the query algorithm satisfies, $\mathbf{dist}(s,t,G) \leq \hat{\mathbf{dist}}(s,t) \leq 2^{O(k\rho)}\mathbf{dist}(s,t,G)$.*

Proof. By the same analysis as in Lemma 8 we can show that all edges $e=(w,v) \in E(H)$, satisfy, $\omega(w,v,H) \geq \mathbf{dist}(w,v,G)$ (where $\omega(x,y,H')$ for nodes x,y and subgraphs H' is the weight of the edge (x,y) in H'). Similarly, $\mu_{i_1} DO^{i_1}_{RZ}(s,p^{i_1}_{j_1}(s)) \geq \mathbf{dist}(s,p^{i_1}_{j_1}(s))$ and $\mu_{i_2} DO^{i_2}_{RZ}(t,p^{i_2}_{j_2}(t)) \geq \mathbf{dist}(t,p^{i_2}_{j_2}(t))$ for every $1 \leq j_1,j_2 \leq k-1, 1 \leq i_1,i_2 \leq \rho$. It thus easily follows that $\hat{\mathbf{dist}}(s,t) \geq \mathbf{dist}(s,t,G)$. It remains to prove that $\hat{\mathbf{dist}}(s,t) \leq 2^{O(k\rho)}\mathbf{dist}(s,t,G)$.

We consider two cases. First suppose that the shortest path $P(s,t,G)$ does not contain any edge in U. Note that in this case $\mathbf{dist}(s,t,G)=\mathbf{dist}(s,t,G_D)$, note also that $\mathbf{Dec}(s,t) \leq 2^{O(k\rho)}\mathbf{dist}(s,t,G)$ and we are done.

The second case is when $P(s,t,G)$ contains at least one edge in U. Let $\{e_i=(x_i,y_i) \mid 1 \leq i \leq r\}$ be the edges in U that appear in $P(s,t)$, where the edge e_{i-1} appears before the edge e_i (namely, the edge e_{i-1} is closer to s in $P(s,t)$ than e_i). In addition, assume also that x_i appears before y_i in $P(s,t)$ (namely, x_i is closer to s in $P(s,t,G)$ than y_i) for every $1 \leq i \leq r$. Note also that $\mathbf{dist}(y_i,x_{i+1},G)=\mathbf{dist}(y_i,x_{i+1},G_D)$.

Let ℓ be the first index such that DO^{ℓ}_{RZ} returns an estimate of $\mathbf{dist}(y_i,x_{i+1})$. Recall that either $DO^{\ell}_{RZ}=\mathbf{dist}(y_i,p^{\ell}_{j_1}(y_i),G_\ell)+\mathbf{dist}(p^{\ell}_{j_1}(y_i),x_{i+1},G_\ell)$ for some j_1 or $DO^{\ell}_{RZ}=\mathbf{dist}(y_i,p^{\ell}_{j_2}(x_{i+1}),G_\ell)+\mathbf{dist}(p^{\ell}_{j_2}(x_{i+1}),x_{i+1},G_\ell)$ for some j_2. Assume w.l.o.g. that $DO^{\ell}_{RZ}(y_i,x_{i+1})=\mathbf{dist}(y_i,p^{\ell}_{j_1}(y_i),G_\ell)+\mathbf{dist}(p^{\ell}_{j_1}(y_i),x_{i+1},G_\ell)$ for some j_1. Recall that by the Thorup-Zwick analysis we have $p^{\ell}_{j_1}(y_i) \in B(x_{i+1})$. Thus by construction the edges $(y_i,p^{\ell}_{j_1}(y_i)),(p^{\ell}_{j_1}(y_i),x_{i+1})$ appear in H, with weights $\mu_\ell\mathbf{dist}(y_i,p^{\ell}_{j_1}(y_i),G_\ell)$ and $\mu_\ell\mathbf{dist}(p^{\ell}_{j_1}(y_i),x_{i+1},G_\ell)$. Thus the graph H contains a path from y_i to x_{i+1} of length $\mu_\ell\mathbf{dist}(y_i,p^{\ell}_{j_1}(y_i),G_\ell)+\mu_i\mathbf{dist}(p^{\ell}_{j_1}(y_i),x_{i+1},G_\ell)=\mu_\ell DO^{\ell}_{RZ}(y_i,x_{i+1})$. By the analysis of the decremental algorithm, we have $\mathbf{dist}(y_i,x_{i+1},H) \leq \mu_\ell DO^{\ell}_{RZ}(y_i,x_{i+1}) \leq 2^{O(k\rho)}\mathbf{dist}(y_i,x_{i+1},G)$. It follows that $\mathbf{dist}(x_1,y_r,H) \leq 2^{O(k\rho)}\mathbf{dist}(x_1,y_r,G)$.

Similarly, let i_1 be the first index such that $DO^{i_1}_{RZ}$ returns an estimate of $\mathbf{dist}(s,x_1)$. Let j_1 be the first index such that $p^{i_1}_{j_1}(s) \in B^{i_1}(x_1)$. Recall that by the analysis of the Thorup-Zwick we have, $\mathbf{dist}(s,p^{i_1}_{j_1}(s),G_{i_1})+\mathbf{dist}(p^{i_1}_{j_1}(s),x_1,G_{i_1}) \leq (4k-3)\mathbf{dist}(s,x_1,G_{i_1})$. By similar analysis of Lemma 8 we have $\mu_{i_1}DO^{i_1}_{RZ}(s,x_1)=\mu_{i_1}(\mathbf{dist}(s,p^{i_1}_{j_1}(s),G_{i_1})+\mathbf{dist}(p^{i_1}_{j_1}(s),x_1,G_{i_1}))=\mu_{i_1}(DO^{i_1}_{RZ}(s,p^{i_1}_{j_1}(s))+DO^{i_1}_{RZ}(p^{i_1}_{j_1}(s),x_1)) \leq 2^{O(k\rho)}\mathbf{dist}(s,x_1,G)$. Note also that the edge $(p^{i_1}_{j_1}(s),x_1)$ exists in H of weight $\mu_{i_1}DO^{i_1}_{RZ}(p^{i_1}_{j_1}(s),x_1)$.

Similarly, we have indices i_2 and j_2 such that $\mu_{i_2}DO^{i_2}_{RZ}(y_r,t)=\mu_{i_2}(DO^{i_2}_{RZ}(t,p^{i_2}_{j_2}(t))+DO^{i_2}_{RZ}(p^{i_2}_{j_2}(t),y_r))$. Note also that the edge $(p^{i_2}_{j_2}(t),y_r)$ exists in H of weight $\mu_{i_2}DO^{i_2}_{RZ}(p^{i_2}_{j_2}(t),y_r)$.

It is not hard to verify now that by concatenating all these paths together with the edges $e_i = (x_i, y_i)$, we have, $\mathbf{dist}(p_{j_1}^{i_1}(s), p_{j_2}^{i_2}(t), H) \leq 2^{O(k\rho)}(\mathbf{dist}(p_{j_1}^{i_1}(s), x_1, G) + \mathbf{dist}(x_1, y_r, G) + \mathbf{dist}(y_r, p_{j_2}^{i_2}(t))$. It follows,

$$\begin{aligned} DO_{TZ,H}(p_{j_1}^{i_1}(s), p_{j_2}^{i_2}(t)) &\leq (2k-1)\mathbf{dist}(p_{j_1}^{i_1}(s), p_{j_2}^{i_2}(t), H) \\ &\leq 2^{O(k\rho)}\mathbf{dist}(p_{j_1}^{i_1}(s), p_{j_2}^{i_2}(t), G). \end{aligned}$$

Hence, we get that,

$$\begin{aligned} \hat{\mathbf{dist}}(s,t) &= \min\{\mu_{i_1} DO_{RZ}^{i_1}(s, p_{j_1}^{i_1}(s)) + DO_{TZ,H}(p_{j_1}^{i_1}(s), p_{j_2}^{i_2}(s)) + \mu_{i_2} DO_{RZ}^{i_2}(t, p_{j_2}^{i_2}(t)) \mid \\ &\quad 1 \leq j_1, j_2 \leq k-1, 1 \leq i_1, i_2 \leq \rho\} \\ &\leq \mu_{i_1} DO_{RZ}^{i_1}(s, p_{j_1}^{i_1}(s)) + (2k-1)\mathbf{dist}(p_{j_1}^{i_1}(s), p_{j_2}^{i_2}(s), H) + \mu_{i_2} DO_{RZ}^{i_2}(t, p_{j_2}^{i_2}(t)) \\ &\leq 2^{O(k\rho)}\mathbf{dist}(s,t,G) \end{aligned}$$

◀

The next lemma bounds the update time.

▶ **Lemma 11.** *The amortized time for a single update is $\tilde{O}(m^{1/2} \cdot n^{2/k})$.*

Proof. The graph H contains $\tilde{O}(|U| \cdot n^{1/k})$ edges and can be constructed in time $\tilde{O}(|U| \cdot n^{1/k})$ given the DO_{RZ} data structures. Constructing $DO_{TZ,H}$ on H takes $\tilde{O}(|U| \cdot n^{1/k} \cdot n^{1/k})$. Recall that after $m^{1/2}$ updates the data structure is being reconstructed so that $|U| \leq m^{1/2}$. Thus the time to construct $DO_{TZ,H}$ in each update step is $\tilde{O}(m^{1/2} \cdot n^{2/k})$.

The total update time of constructing the decremental data structure **Dec** is $\tilde{O}(mn^{1/k})$. This should be amortized over the $m^{1/2}$ updates until **Dec** is being reconstructed again. We get that the amortized per update for maintaining **Dec** is $\tilde{O}(m^{1/2}n^{1/k})$. It is not hard to see now that the amortized update time is $\tilde{O}(m^{1/2} \cdot n^{2/k})$.

The lemma follows. ◀

The next lemma bounds the query time.

▶ **Lemma 12.** *The query time is $O((k\rho)^2)$.*

Proof. Recall that the query algorithm returns the minimum distance between $\mathbf{Dec}(s,t)$ and the minimum

$$\min\{\mu_{i_1} DO_{RZ}^{i_1}(s, p_{j_1}^{i_1}(s)) + DO_{TZ,H}(p_{j_1}^{i_1}(s), p_{j_2}^{i_2}(s)) + \mu_{i_2} DO_{RZ}^{i_2}(t, p_{j_2}^{i_2}(t)) \mid 1 \leq j_1, j_2 \leq k-1, 1 \leq i_1, i_2 \leq \rho\}.$$

Computing $\mathbf{Dec}(s,t)$ takes $O(k\rho)$ time. Computing $\mu_{i_1} DO_{RZ}^{i_1}(s, p_{j_1}^{i_1}(s))$ for $1 \leq j_1 \leq k-1$ and $1 \leq i_1 \leq \rho$ takes $O(1)$ (the algorithm stores $DO_{RZ}^{i_1}(s, p_{j_1}^{i_1}(s)) = \mathbf{dist}(s, p_{j_1}^{i_1}(s), G_{i_1})$). Thus computing all $DO_{RZ}^{i_1}(s, p_{j_1}^{i_1}(s))$ for $1 \leq j_1 \leq k-1$ and $1 \leq i_1 \leq \rho$ takes $O(k\rho)$ time. Similarly, computing all $DO_{RZ}^{i_2}(t, p_{j_2}^{i_2}(2))$ for $1 \leq j_2 \leq k-1$ and $1 \leq i_2 \leq \rho$ takes $O(k\rho)$ time.

In addition, Thorup and Zwick query algorithm is $O(k)$. It was shown in [7] how to reduce the query time to constant (while keeping the rest of the parameters the same). Thus, computing $DO_{TZ,H}(x,y)$ for two nodes x, y can be done in $O(1)$ time.

It is not hard to see now that finding the minimum takes $O(k^2\rho^2)$ time. ◀

By maintaining this data structure for setting $k' = \lfloor k/2 \rfloor$ we can get update time $\tilde{O}(m^{1/2} \cdot n^{1/k})$. This will increase the logarithm of the stretch by only an $O(1)$ factor, and the query time goes up by $O(1)$. This proves Theorem 1.

References

1 G. Ausiello, G. F. Italiano, A. Marchetti-Spaccamela, and U. Nanni. Incremental algorithms for minimal length paths. *J. Algorithms, 12(4)*, 615–638, 1991.

2 S. Baswana, R. Hariharan, and S. Sen. Improved decremental algorithms for transitive closure and all-pairs shortest paths. In *34th ACM Symp. on Theory of Computing (STOC)*, 117–123, 2002.

3 S. Baswana, R. Hariharan, and S. Sen. Maintaining all-pairs approximate shortest paths under deletion of edges. In *14th ACM Symp. on Discrete Algorithms (SODA)*, 394–403, 2003.

4 S. Baswana, S. Khurana, and S. Sarkar. Fully dynamic randomized algorithms for graph spanners. In *ACM Transactions on Algorithms*, 8(4):35, 2012.

5 A. Bernstein. Fully dynamic approximate all-pairs shortest paths with query and close to linear update time. In *50th IEEE Symp. on Foundations of Computer Science (FOCS)*, 50–60, 2009.

6 A. Bernstein, L. Roditty. Improved dynamic algorithms for maintaining approximate shortest paths under deletions. In *22nd ACM Symp. on Discrete Algorithms (SODA)*, 1355–1365, 2011.

7 S.Chechik. Approximate Distance Oracles with Constant Query Time. To appear in *Proc. 46th ACM Symp. on Theory of Computing (STOC)*, 2014.

8 C. Demetrescu and G. F. Italiano. A new approach to dynamic all pairs shortest paths. *J. of the ACM, 51*, 2004.

9 C. Demetrescu and G. F. Italiano. Fully dynamic all pairs shortest paths with real edge weights. *J. of Computer and System Sciences, 72(5)*, 813–837, 2006. Special issue of *FOCS*'01.

10 Y. Dinitz. Dinitz' algorithm: The original version and Even's version. In *Essays in Memory of Shimon Even*, 218–240, 2006

11 M. R. Henzinger and V. King. Maintaining Minimum Spanning Forests in Dynamic Graphs. *SIAM J. Computing, 31(2)*, 364–374, 2001.

12 M. R. Henzinger, S. Krinninger and D. Nanongkai. A Subquadratic-Time Algorithm for Decremental Single-Source Shortest Paths. In *25th ACM Symp. on Discrete Algorithms (SODA)*, 1053–1072, 2014.

13 V. King. Fully dynamic algorithms for maintaining all-pairs shortest paths and transitive closure in digraphs. In *40th IEEE Symp. on Foundations of Computer Science (FOCS)*, 81–91, 1999.

14 V. King and M. Thorup. A space saving trick for directed dynamic transitive closure and shortest path algorithms. In Jie Wang, editor, COCOON, volume 2108 of *Lecture Notes in Computer Science*, 268–277, 2001.

15 M. Thorup. Fully-dynamic all-pairs shortest paths: faster and allowing negative cycles. In *9th Scandinavian Workshop on Algorithm Theory (SWAT)*, 384–396, 2004.

16 M. Thorup. Worst-case update times for fully-dynamic all-pairs shortest paths. In *37th ACM Symp. on Theory of Computing (STOC)*, 112–119, 2005.

17 M. Thorup and U. Zwick. Compact routing schemes. In *13th ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, 1–10, 2001.

18 M. Thorup and U. Zwick. Approximate distance oracles. In *Journal of the ACM*, pages 1–24, 2005.

19 L. Roditty and U. Zwick. Dynamic approximate all-pairs shortest paths in undirected graphs. In *45th IEEE Symp. on Foundations of Computer Science (FOCS)*, 499–508, 2004.

20 L. Roditty and U. Zwick. Dynamic Approximate All-Pairs Shortest Paths in Undirected Graphs. *SIAM J. Comput.* 41(3): 670–683, 2012.

21 L. Roditty and U. Zwick. On dynamic shortest paths problems. In *Proc. 18th European Symposium on Algorithms (ESA)*, 580–591, 2004.

Approximation Algorithms for Minimum-Load k-Facility Location

Sara Ahmadian[*1], Babak Behsaz[2], Zachary Friggstad[2], Amin Jorati[2], Mohammad R. Salavatipour[†2], and Chaitanya Swamy[‡1]

1 **Department of Combinatorics and Optimization, University of Waterloo**
Waterloo, ON, Canada, N2L 3G1
{sahamdian,cswamy}@math.uwaterloo.ca

2 **Department of Computing Science, University of Alberta**
Edmonton, AB, Canada, T6G 2E8
{behsaz,zacharyf,jorati,mrs}@ualberta.ca

Abstract

We consider a facility-location problem that abstracts settings where the cost of serving the clients assigned to a facility is incurred by the facility. Formally, we consider the *minimum-load k-facility location* (MLkFL) problem, which is defined as follows. We have a set $\mathcal{F}$ of facilities, a set $\mathcal{C}$ of clients, and an integer $k \geq 0$. Assigning client j to a facility f incurs a connection cost $d(f,j)$. The goal is to open a set $F \subseteq \mathcal{F}$ of k facilities, and assign each client j to a facility $f(j) \in F$ so as to minimize $\max_{f \in F} \sum_{j \in \mathcal{C}: f(j)=f} d(f,j)$; we call $\sum_{j \in \mathcal{C}: f(j)=f} d(f,j)$ the *load* of facility f. This problem was studied under the name of min-max star cover in [6, 2], who (among other results) gave bicriteria approximation algorithms for MLkFL for when $\mathcal{F} = \mathcal{C}$. ML$k$FL is rather poorly understood, and only an $O(k)$-approximation is currently known for MLkFL, *even for line metrics*.

Our main result is the *first polynomial time approximation scheme* (PTAS) for MLkFL on line metrics (note that no non-trivial true approximation of any kind was known for this metric). Complementing this, we prove that MLkFL is strongly *NP*-hard on line metrics. We also devise a quasi-PTAS for MLkFL on tree metrics. MLkFL turns out to be surprisingly challenging even on line metrics, and resilient to attack by the variety of techniques that have been successfully applied to facility-location problems. For instance, we show that: (a) even a configuration-style LP-relaxation has a bad integrality gap; and (b) a multi-swap k-median style local-search heuristic has a bad locality gap. Thus, we need to devise various novel techniques to attack MLkFL.

Our PTAS for line metrics consists of two main ingredients. First, we prove that there always exists a near-optimal solution possessing some nice structural properties. A novel aspect of this proof is that we first move to a mixed-integer LP (MILP) encoding the problem, and argue that a MILP-solution minimizing a certain potential function possesses the desired structure, and then use a rounding algorithm for the generalized-assignment problem to "transfer" this structure to the rounded integer solution. Complementing this, we show that these structural properties enable one to find such a structured solution via dynamic programming.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases approximation algorithms, min-max star cover, facility location, line metrics

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.17

[*] Supported by the last author's NSERC grant 327620-09.
[†] Supported by NSERC.
[‡] Supported in part by NSERC grant 327620-09, an NSERC Discovery Accelerator Supplement Award, and an Ontario Early Researcher Award.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 17–33

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

1 Introduction

Facility-location (FL) problems have been widely studied in the Operations Research and Computer Science communities (see, e.g., [13] and the survey [16]), and have a wide range of applications. These problems are typically described in terms of an underlying set of clients that require service, and a candidate set of facilities that provide service to these clients. The goal is to determine which facilities to open, and decide how to assign clients to open facilities to minimize some combination of the facility-opening and client-connection (a.k.a service) costs. An oft-cited prototypical example is that of a company wanting to decide where to locate its warehouses/distribution centers so as to serve its customers in a cost-effective manner.

We consider settings where the cost of serving the clients assigned to a facility is incurred by the facility; for instance, in the above example, each warehouse may be responsible for supplying its clients and hence bears a cost equal to the total cost of servicing its clients. In such settings, it is natural to consider the problem of minimizing the maximum cost borne by any facility. Formalizing this, we consider the following mathematical model. We are given a set $\mathcal{F}$ of facilities, a set $\mathcal{C}$ of clients, and an integer $k \geq 0$. Assigning client j to a facility f incurs a *connection* or *service cost* $d(f, j)$. There are no facility-opening costs. The goal is to open k facilities from $\mathcal{F}$ and assign each client j to an open facility $f(j)$ so as to minimize the maximum *load* of an open facility, where the load of an open facility f is defined as $\sum_{j \in \mathcal{C}: f(j)=f} d(f, j)$; that is, the load of f is the total connection cost incurred for the clients assigned to it. We call this the *minimum-load k-facility location* (MLkFL) problem. As is common in the study of facility-location problems, we assume that the clients and facilities lie in a common metric space, so the $d(f, j)$s form a metric.

Despite the extensive amount of literature on facility-location problems, there is surprisingly little amount of work on MLkFL and it remains a rather poorly understood problem (see [15]). One can infer that the problem is *NP*-hard, even when the set of open facilities is fixed, via a reduction from the makespan-minimization problem on parallel machines, and that an $O(k)$-approximation can be obtained by running any of the various $O(1)$-approximation algorithms for k-*median* [4, 9, 8, 3, 12] (where one seeks to minimize the *sum* of the facility loads). No better approximation algorithms are known for MLkFL *even on line metrics*, and this was mentioned as an open problem in [15]. The only work on approximation algorithms for this problem that we are aware of is due to Even et al. [6] and Arkin et al. [2], who refer to this problem as *min-max star cover* (where $\mathcal{F} = \mathcal{C}$). [1] Both works obtain *bicriteria* approximation algorithms for MLkFL in general metrics, which means that the algorithm returns a solution with near-optimal maximum load but may need to open more than k facilities. For MLkFL on star metrics and when $\mathcal{F} = \mathcal{C}$, some $O(1)$-approximation algorithms follow from work on minimum-makespan scheduling and [6, 2] (see "Related work").

1.1 Our Results

We completely resolve the status of min-load k-FL on line metrics. As we elaborate below (see "Our Techniques"), MLkFL turns out to be surprisingly challenging even on line metrics, and seems resilient to attack by the variety of techniques that have been successfully applied to facility-location problems, including LP-rounding, local search, and primal-dual methods. Our main result is that despite these difficulties, one can devise a polynomial-time approximation

[1] Jorati [10], in his Master's thesis, obtained a preliminary version of some of our current results.

scheme (PTAS) for MLkFL on line metrics (Theorem 1). As mentioned earlier, this is the *first* approximation algorithm for MLkFL on line metrics that achieves anything better than an $O(k)$-approximation.

We also consider MLkFL in tree metrics (Section 4). First, we observe that the quasi-PTAS obtained by Jorati [10] for line metrics extends to yield a quasi-PTAS (QPTAS) for tree metrics (Theorem 9). Next, we consider the special case of star metrics, but in the more-general setting where clients may have non-uniform integer demands $\{D_j\}_{j\in\mathcal{C}}$ and the demand of a client may be split *integrally* between several open facilities. We now define the load of a facility f to be $\sum_j x_{fj}d(f,j)$, where $x_{fj} \in \mathbb{Z}_{\geq 0}$ is the amount of j's demand that is assigned to f. We devise a 14-approximation algorithm for MLkFL on star metrics with non-uniform demands (Theorem 10). Notice that when we restrict the metric to be a star metric, we cannot create colocated copies of a client (without destroying the star topology), which makes the setting with non-uniform demands strictly more general than the unit-demand setting.

In Section 5, we obtain various computational-complexity and integrality-gap lower bounds for MLkFL. Complementing our PTAS, we show (Theorem 11) that MLkFL is *strongly* NP*-hard* on line metrics (and hence, a PTAS is the best approximation that one can hope to achieve in polytime unless $P = NP$). We also show that MLkFL is APX-hard in the Euclidean plane (Theorem 12). Finally, we justify our comment about the difficulty of tackling MLkFL via the various LP-based methods developed for facility-location problems by showing that even a configuration-style LP-relaxation for MLkFL—where we "guess" the optimum value B and have a variable $x_{f,S}$ for every facility f and every possible set S of clients such that $\sum_{j\in S} d(f,j) \leq B$—has an integrality gap of $\Omega(k/\log k)$ even for line metrics (Theorem 13). Note that the configuration LP is stronger than the natural LP-relaxation for MLkFL. Moreover, this holds even if the graph consisting of the edges (j,f) such that $d(j,f) \leq B$—call these feasible edges—is connected. This is in contrast with capacitated k-center [5, 1], where a large integrality gap for the natural LP arises due to the fact that the graph of feasible edges is disconnected.

1.2 Our Techniques

Before detailing the techniques underlying our PTAS for line metrics, we describe some of the difficulties encountered in applying the machinery developed for (other) facility-location problems to MLkFL (even on line metrics). One prominent source of techniques for facility location are LP-based methods. However, our integrality-gap lower bound for line metrics points to the difficulty in leveraging such LP-based insights. In fact, we do not know of any LP-relaxation for MLkFL with a constant integrality gap even on line metrics. An approach that often comes to the rescue for FL problems when there is no known good LP-relaxation (e.g., capacitated FL) is local search, however the min-max nature of MLkFL makes it difficult to exploit this. In particular, one can come up with simple examples where a k-median style multi-swap local-search does not yield any bounded approximation ratio even for line metrics. Given these difficulties, one needs to find new venues of attack for MLkFL. Our PTAS for line metrics consists of two main ingredients. First, we prove that there always exists a near-optimal solution possessing some nice structural properties (Section 3.1). Namely, the collection of intervals corresponding to "small" client assignments forms a laminar family. We prove this by "fractionally uncrossing" the small client assignments while preserving the loads at each facility, so the resulting fractional assignment does not contain large strictly fractional assignments. This solution is then rounded using the rounding algorithm of [17] for the *generalized assignment problem* (GAP), and this rounding procedure preserves the laminarity property for small assignments.

Second, we show in Section 3.2 that these structural properties enable one to find such a structured solution via dynamic programming (DP). Roughly speaking, the DP pieces together solutions to subproblems in a way that corresponds to the tree-like structure of a laminar family. To handle the unstructured large client assignments, the DP carries enough information about how large clients cross the boundary of the subproblem being considered.

1.3 Related Work

There is a wealth of literature on facility-location problems (see, e.g., [13, 16]); we limit ourselves to the work that is relevant to MLkFL. As mentioned earlier, Even et al. [6] and Arkin et al. [2] are the only two previous works that study MLkFL (under the name min-max star cover). They view the problem as one where we seek to cover the nodes of a graph by stars (hence the name min-max star cover), and obtain bicriteria guarantees. Viewed from this perspective, MLkFL falls into the class of problems where we seek to cover nodes of an underlying graph using certain combinatorial objects. Even et al. and Arkin et al. consider various other min-max problems—where the number of covering objects is fixed and we seek to minimize the maximum cost of an object—in this genre. Both works devise a 4-approximation algorithm when the covering objects are trees (see also [14]), and Even et al. obtain the same approximation for the rooted problem where the roots of the trees are fixed. Arkin et al. obtain an $O(1)$-approximation when the covering objects are paths or walks. The approximation guarantees for min-max tree cover were improved by Khani and Salavatipour [11]. All of these works also consider the version of the problem where we fix the maximum cost of a covering object and seek to minimize the number of covering objects used.

For MLkFL on star metrics, when $\mathcal{F} = \mathcal{C}$, certain results follow from some known results and the above min-max results. For example, it is not hard to show that MLkFL, even with non-unit demands, can be reduced to the makespan-minimization problem on parallel machines while losing a factor of 2.[2] Since the latter problem admits a PTAS [7], this yields a $(2+\epsilon)$-approximation algorithm for MLkFL on star metrics when $\mathcal{F} = \mathcal{C}$. When $\mathcal{F} = \mathcal{C}$ and with unit demands, one can also infer that (for star metrics) the objective value of any solution for min-max tree cover (viewed in terms of the node-sets of the trees) is within a constant factor of its objective value for min-max star cover. (This is simply because for any set S of nodes, the cost of the *best* star spanning S is at most twice the cost of the minimum spanning tree for S.) These correspondences however break down when $\mathcal{F} \neq \mathcal{C}$, even for unit demands. Our 14-approximation algorithm for star metrics works for arbitrary $\mathcal{F}, \mathcal{C}$ sets *and* non-unit (equivalently, non-uniform) demands.

As with the k-median and k-center problems, MLkFL can also be motivated and viewed as a clustering problem: we seek to cluster points in a metric space around k centers, so to minimize the maximum load (or "star cost") of a cluster. Whereas MLkFL and k-center are min-max clustering problems, where the quality is measured by the *maximum* cost (under some metric) of a cluster, k-median is a min-sum clustering problem, where the clustering quality is measured by summing the cost of each cluster.

Finally, observe that if we fix the set of k open facilities, then the problem of determining the client assignments is a special case of GAP. There is a well-known 2-approximation

[2] If we require that all k facilities lie at the root r of the star, then the resulting problem is precisely a makespan-minimization problem on k parallel machines. Given a partition $C_1, \ldots, C_k$ of the client-set obtained by solving this problem, we can simply open, for each C_i, a facility at the node in C_i that is closest to r. This increases the maximum load by a factor of at most 2.

algorithm for GAP [17]. As noted earlier, this algorithm plays a role in the *analysis* of our PTAS for line metrics (but not the algorithm itself), when we reason about the existence of well-structured near-optimal solutions.

2 Problem Definition

In the minimum-load k-facility location (MLkFL) problem, we are given a set of clients $\mathcal{C}$ and a set of facilities $\mathcal{F}$ in a given metric space d. The distance between any pair of points $i, j \in \mathcal{C} \cup \mathcal{F}$ is denoted by $d(i, j)$. Additionally we are given an integer $k \geq 1$. The goal is to select k facilities $f_1, \ldots, f_k$ to open and assign each client j to an open facility so as to minimize $\max_{i=1}^{k} \sum_{j \in \mathcal{C}: f(j)=i} d(i, j)$, where $f(j)$ is the facility to which client j is assigned. We use the terms facility and center interchangeably. We frequently use the term star to refer to a pair (f, S), where f is an open facility in the solution and $S \subseteq \mathcal{C}$ is the collection of clients assigned to f; we also refer to f as the center of this star. The cost of this star, which is the load of facility f, is $\sum_{j \in S} d(f, j)$. Thus, our goal is to find k stars, $(f_1, S_1), (f_2, S_2), \ldots, (f_k, S_k)$, centered at facilities so that they "cover" all the clients (i.e. $\mathcal{C} = \cup_{i=1}^{k} S_i$) and the maximum load of a facility (or cost of the star) is minimized. Throughout, we use OPT to denote an optimum solution and L^{opt} to denote its cost.

3 A PTAS for Line Metrics

In this section we focus on MLkFL on line metrics and prove Theorem 1. Here, each client/facility $i \in \mathcal{C} \cup \mathcal{F}$ is located at some rational point $v_i \in \mathbb{R}$. It may be that $v_i = v_j$ for $i \neq j$, for instance when we have collocated clients. To simplify notation we use the term "point" to refer to a client or facility $i \in \mathcal{C} \cup \mathcal{F}$ as well as to its location v_i. The distance $d(i, j)$ between points $i, j \in \mathcal{C} \cup \mathcal{F}$ is simply $|v_i - v_j|$. We assume that $|\mathcal{C} \cup \mathcal{F}| = n$ and that $0 \leq v_1 \leq v_2 \leq \ldots \leq v_n$. For a star (f, S) in a MLkFL solution and for any $v \in S$, say that the open interval with endpoints f and v is an *arm* of the star (f, S) and we say that f covers v. For $S' \subseteq S$, we sometimes use the phrase "load of f by S'" to refer to the sum of the lengths of arms of f to the clients in S'. The main result of this section is the following theorem.

▶ **Theorem 1.** *There is a $(1 + \varepsilon)$-approximation algorithm for* MLkFL *on line metrics for any constant $0 < \varepsilon \leq 1$.*

Our high-level approach is similar to other min-max problems. Namely, we present an algorithm that, given a guess B on the optimum solution value, will either certify that $B < L^{opt}$ or else find a solution with cost not much more than B. Our main technical result, which immediately yields Theorem 1 is the following.

▶ **Theorem 2.** *Let $\Pi = (\mathcal{C} \cup \mathcal{F}, d, k)$ be a given* MLkFL *instance on a line metric. For any constant $0 < \epsilon \leq 1$ and any $B \geq 0$, there is a polynomial-time algorithm $\mathcal{A}$ that either finds a feasible solution with cost at most $(1 + 18\epsilon) \cdot B$ or declares that no feasible solution with cost at most B exists. If $B \geq L^{opt}$, then it always finds a feasible solution with cost at most $(1 + 18\epsilon) \cdot B$.*

We show how to complete the proof of Theorem 1, assuming Theorem 2 is true.

Proof. Set $\epsilon := \varepsilon/18$. We use binary search to find a value $B \leq L^{opt}$ such that algorithm $\mathcal{A}$ from Theorem 2 finds a solution with cost $\leq (1 + 18\epsilon) \cdot B \leq (1 + 18\epsilon) \cdot L^{opt}$. Return this solution.

Since the points v_i are rational and since $n \cdot v_n$ is clearly an upper bound on the optimum solution, then we may perform the binary search over integers $\alpha \in [0, nv_n\Delta]$ where Δ is such that $v_i\Delta \in \mathbb{Z}$ for each point i. For each such value α in the binary search, we try algorithm $\mathcal{A}$ with value $B = \frac{\alpha}{\Delta}$. ◀

In what follows, we describe algorithm $\mathcal{A}$. We will assume that $B \geq L^{opt}$ and show how to find a solution with cost at most $(1 + 18\epsilon) \cdot B$. Let $\mathcal{S}_B$ denote a collection of stars $\{(f_1, S_1), \ldots, (f_k, S_k)\}$ with cost at most B. In the remainder of this section, we will describe some preprocessing steps that simplify the structure of the problem. In Section 3.1 we prove that a well-structured near-optimum solution exists and in Section 3.2 we describe a dynamic programming algorithm that finds such a near-optimum well-structured solution.

Without loss of generality, we assume that $1/\epsilon$ is an integer. We start with some preprocessing steps. Note that $d(i, f) \leq B$ for any $i \in S$ of a star (f, S) in $\mathcal{S}_B$. So, if the distance of two consecutive points on the line is more than B then we can decompose an instance into instances where the distance of any two consecutive points is at most B. For each of the resulting instances Π', we find the smallest k' such that running the subsequent algorithm on the instance with k' instead of k finds a solution with cost at most $(1 + 18\epsilon)B$. Since we are assuming $B \geq L^{opt}$, then the sum of these k' values over the subinstances is at most k. Note that in each subinstance Π' we can assume $0 \leq v_i \leq n \cdot B$ for each point v_i.

Next, we perform a standard scaling of distances. Move every point $i \in \mathcal{C} \cup \mathcal{F}$ left to its nearest integer multiple of $\frac{\epsilon B}{n}$ and then multiply this new point by $\frac{n}{\epsilon B}$. That is, move i from v_i to $\lfloor v_i \cdot n/\epsilon B \rfloor$. Denote the new position of client/facility i by v'_i. The following describes how the optimum solutions to the original and new locations relate.

▶ **Lemma 3.** *The optimum solution has cost at most $(1 + 1/\epsilon) \cdot n$ in the instance given by the new positions v'. Furthermore, any solution with cost at most $(1 + \alpha\epsilon) \cdot (1 + 1/\epsilon) \cdot n$ for the new positions has cost at most $(1 + (2 + 2\alpha)\epsilon) \cdot B$ in the original instance.*

Proof. After sliding each point v_i left to its nearest integer multiple of $\frac{\epsilon B}{n}$, the distance between any two points changes by at most $\frac{\epsilon B}{n}$. Therefore, the load of any star changes by at most ϵB so each star has load at most $(1 + \epsilon)B$. Finally, after multiplying all points by $\frac{n}{\epsilon B}$ we have that the maximum load of any star is at most $(1 + 1/\epsilon) \cdot n$.

Now consider any solution with cost at most $(1 + \alpha \cdot \epsilon) \cdot (1 + 1/\epsilon) \cdot n$. Scaling the points v' back by $\epsilon B/n$ produces a solution with cost at most $(1 + \alpha \cdot \epsilon)(1 + \epsilon) \cdot \epsilon B \leq (1 + (1 + 2\alpha)\epsilon) \cdot B$. Then sliding, any two points i, j back to their original positions v_i, v_j changes their distance by at most $\epsilon B/n$, so doing this for all points changes the cost of any star by at most ϵB. The resulting stars then have cost at most $(1 + (2 + 2\alpha)\epsilon) \cdot B$. ◀

In subsequent sections, we describe a $(1+8\epsilon)$-approximation for any one of the subinstances Π' of Π, except we use the new points v'_i. By Lemma 3, this gives us a solution to Π with cost at most $(1 + 18\epsilon)B$, proving Theorem 2. To simplify notation, we use v_i to refer to the *new* location of point $i \in \mathcal{C} \cup \mathcal{F}$ (i.e. rename v'_i to v_i). Similarly, the notation $d(i, j)$ for $i, j \in \mathcal{C} \cup \mathcal{F}$ refers to these new distances $|v'_i - v'_j|$ and B denotes the new budget $(1 + 1/\epsilon) \cdot n$. From now on, we assume our given instance Π of MLkFL satisfies the following properties: a) Each point v_i is an integer between 0 and $(1 + 1/\epsilon) \cdot n^2$, b) There is a solution $\mathcal{S}_B$ with cost at most $B = (1 + 1/\epsilon) \cdot n$.

3.1 Structure of Near Optimum Solutions

In this section, we show that there is a near-optimum solution to the instance Π with clients and facilities $\mathcal{C} \cup \mathcal{F}$ that has some suitable structural properties. In Section 3.2, we will find such a solution using a dynamic programming approach.

We denote the open interval between two points v_i and v_j on the line by $I_{i,j}$ and call this the *arm* between i and j (assuming that one of i, j is a client and the other is a facility). An arm $I_{i,j}$ is *large* if $d(i,j) > \epsilon B$ and is *small* otherwise. We say that two arms $I_{i,j}$ and $I_{i',j'}$ *cross* if $I_{i,j}$ is not contained in $I_{i',j'}$ or vice versa, and $I_{i,j} \cap I_{i',j'} \neq \emptyset$.

A *well-formed solution* for an MLkFL instance is a solution in which the small arms between clients and their assigned facilities (centers) do not cross. We show that there exists a low cost well-formed solution in two steps. First, we demonstrate the existence of a *fractional solution* where there are k (integral) facilities and the clients are assigned to these centers fractionally. This will be such that the fractional load of each facility is still at most B, all strictly fractional arms in the support have length at most $2\epsilon B$, and that all small arms in the support of the solution do not cross.

Second, we use a rounding algorithm for the Generalized Assignment Problem (GAP) by Shmoys and Tardos [17] to round such a fractional solution to an integral solution with cost at most $(1 + 2\epsilon)B$. We emphasize that that this rounding algorithm is not a part of our algorithm, it is only used to demonstrate the existence of a well-structured solution.

For the first step, we will consider a fractional uncrossing argument to eliminate crossings. Instead of proving the fractional uncrossing process eventually terminates, we will instead provide a potential function that strictly decreases in a fractional uncrossing. This potential function is the objective function of a mixed integer-linear program below; thus an optimal solution will not contain any crossings between small arms its support.

We let $C_B = \{f_1, \ldots, f_k\}$ denote the centers (facilities) of the stars in the solution $\mathcal{S}_B$ (recall that each star in C_B has cost/load at most B). The variable x_{ij} indicates that client j is assigned to facility $f_i \in C_B$. The first constraint ensures every client is assigned to some facility and the second ensures the cost of a star (i.e. load of a facility) does not exceed B.

We stress that this is *not* a relaxation for MLkFL. The objective function is more similar to the objective function for the k-median problem. Rather, we will only be using this to help demonstrate the existence of a well-formed solution. The objective function acts as a potential function.

$$\begin{aligned}
\text{minimize} \quad & \sum_{f_i \in C_B} \sum_{j \in \mathcal{C}} d(f_i, j) \cdot x_{ij} && && \text{(MIP)}\\
\text{subject to} \quad & \sum_{f_i \in C_B} x_{ij} = 1 && \forall j \in \mathcal{C} \\
& \sum_{j \in V} d(f_i, j) \cdot x_{ij} \leq B && \forall f_i \in C_B \\
& x_{ij} \in \{0,1\} && \forall i, j : d(f_i, j) \geq 2\epsilon B \\
& 0 \leq x_{ij} \leq 1 && \forall i, j : d(f_i, j) < 2\epsilon B.
\end{aligned}$$

▶ **Lemma 4.** *There is a feasible solution x to mixed integer-linear program* (MIP) *where the small arms in the support of x do not cross.*

Proof (Sketch). First observe that there is in fact a feasible solution x because the integer solution $\mathcal{S}_B$ is feasible for this ILP. By standard theory of mixed-integer programming and the fact that the set of feasible solutions is bounded, there is then an optimal solution x. We claim that no two small arms in the support of an optimal solution to (MIP) are crossing. The low-level details that support this claim will appear in the full version, but the idea is that if two small arms cross then we can fractionally uncross them to get a strictly better solution to (MIP) and the new fractional arms have length at most $2\epsilon B$. ◀

We will use Lemma 4 to prove the existence of a near-optimum solution to instance Π where the small arms used by clients do not cross. To complete this proof, we rely on a structural result concerning the polytope of a relaxation for the following scheduling problem.

▶ **Definition 5.** In the scheduling problem on unrelated machines, we are given machines $m_1, \ldots, m_k$, jobs $j_1, \ldots, j_n$, and processing times $p(m_i, j_a) \geq 0$ between any job j_a and any machine m_i. The goal is to assign each job j_a to a machine $\phi(j_a) \in \{m_1, \ldots, m_k\}$ to minimize the maximum total running time $\sum_{a:\phi(j_a)=m_i} p(m_i, j_a)$ of any machine.

Shmoys and Tardos prove a result concerning the polytope of an LP relaxation for this problem, as a part of a more general result concerning the related *Generalized Assignment Problem* (GAP). The following summarizes the results they obtain that are relevant for our work.

▶ **Theorem 6** (Shmoys and Tardos, [17])**.** *Suppose we have a bound B and fractional values $x(m_i, j_a) \geq 0$ for each job j_a and each machine m_i that satisfy the following:*

- $\sum_{i=1}^{k} x(m_i, j_a) = 1$ *for each job* j_a*,*
- $\sum_{a=1}^{n} p(m_i, j_a) x(m_i, j_a) \leq B$ *for each machine* m_i*.*

Then there is an assignment ϕ of jobs to machines such that $x(\phi(j_a), j_a) > 0$ for each job j_a and the maximum load of any machine under ϕ is at most $B + \max_{a,i:0<x(m_i,j_a)<1} p(m_i, j_a)$.

We use the above theorem together with Lemma 4 to prove the following.

▶ **Theorem 7.** *There is a feasible (integer) solution to the* MLkFL *instance* Π *with maximum load $(1 + 2\epsilon)B$ on each star such that no two small arms cross.*

Proof. Let x^* be the fractional solution provided by Lemma 4. We view x^* as a solution to the following scheduling problem on unrelated machines. We have k machines $m_1, \ldots, m_k$, each corresponding to a facility $f_i \in C_B$. For each client $a \in \mathcal{C}$, there is a single job j_a. The processing time $p(m_i, j_a)$ of job j_a on machine m_i is $|v_i - v_a|$, the distance between the corresponding locations.

Now, x^* fractionally assigns each job j_a to the machines to a total extent of 1 and the maximum (fractional) load at machine m_i is B. Furthermore, the only strictly fractional assignments (i.e. those with $0 < x_{ij} < 1$) have $|v_i - v_j| \leq 2\epsilon B$. In the scheduling terminology, the only strictly fractional assignments are between a job j_a and a machine m_i such that $p(m_i, j_a) \leq 2\epsilon B$.

Theorem 6 shows we can transform this fractional assignment x^* into an integer assignment such that a) if client j is assigned to facility/center i, then $x^*_{ij} > 0$ and b) the maximum load of a facility is $B + \max_{i,j:0<x^*_{ij}<1} |v_i - v_j| \leq B + 2\epsilon B$. In this solution, small arms used by clients do not cross because they come from the support of x^*. ◀

▶ Remark. Our distinction between small and large arms is not just for the sake of obtaining a PTAS. In fact, we do not know if there is a completely uncrossed, $O(1)$-approximate solution. For instance, we have an example where iterating the fractional uncrossing argument to uncross all arms may create a fractional arm whose length is longer than B by a super-constant factor.

3.2 Finding a Well-Formed Solution

3.2.1 Step Min-max Cost

Theorem 7 shows that there is a solution of cost at most $(1 + 2\epsilon)B$ such that no two small arms (i.e. length $\leq \epsilon B$) used to assign clients to centers cross. Call this solution $\mathcal{S}'_{\mathcal{B}}$. We now show that we can find such a well-structured solution of cost at most $(1 + 8\epsilon)B$.

The main idea behind our approach is the following. If it were true that a near-optimum solution did not have any crossing arms (large or small) then we can find such a solution using a dynamic programming approach. At a very high-level, we could exploit the laminar structure of the solution by decomposing the solution into a family of nested intervals $\mathcal{I}$ such that for every $I \in \mathcal{I}$ there is one center c with $v_c \notin I$ such that clients in I are served either by centers in I or by c. From this, we can consider triples (I, c, r) where $I \in \mathcal{I}$, c is a location outside of I and r is some integer between 0 and $\text{poly}(n, 1/\epsilon)$ describing the load assigned to c from clients in I. We can look for partial solutions parametrized by these triples and relate them through an appropriate recurrence.

Unfortunately, we are only guaranteed that the small arms do not cross in our near-optimum solution so the collection of all arms in the solution is not necessarily laminar. To handle this general case, we must carry extra information through our dynamic programming approach. We begin by coarsening how we measure the length of long arms.

First, recall that all long arms have length more than ϵB. Thus, each facility is serving at most $\frac{(1+2\epsilon)B}{\epsilon B} \leq \frac{3}{\epsilon}$ clients that are at distance more than ϵB; in other words each star is assigned at most $\frac{3}{\epsilon}$ long arms in the solution provided by Theorem 7. Say that one such long arm is between client j and center i. If we moved both j and i left to their nearest integer multiples of $\epsilon^2 B$, then their distance changes by at most $\epsilon^2 B$. If this is done for all long arms assigned to a center i, then the total load of center i due to long arms changes by at most $3\epsilon B$.

Now, notice that this way to measure the distance between client j and center i is simply $\epsilon^2 B$ times the number of integer multiples of $\epsilon^2 B$ that lie in the half-open interval $(v_i, v_j]$ if $v_i < v_j$ or $(v_j, v_i]$ if $v_j < v_i$. In the dynamic programming algorithm described below, we will use this coarse method to measure the distance of long arms and call this the *perceived cost* of the star. More specifically, the perceived cost of a star (f, S) is the total cost of the small arms plus $\sum_{j \in S: I_{f,j} \text{ long}} |v''_f - v''_j|$ where v''_i is the nearest multiple of $\epsilon^2 B$ to the left of v_i. The following is proved using arguments similar to the proof of Lemma 3, recalling that every star in $\mathcal{S}'_{\mathcal{B}}$ has at most $3/\epsilon$ long arms.

▶ **Lemma 8.** *The perceived cost of every star in $\mathcal{S}'_{\mathcal{B}}$ is at most $(1 + 5\epsilon)B$. Furthermore, any star with perceived cost at most $(1 + 5\epsilon)B$ and at most $3/\epsilon$ long arms has (actual) cost at most $(1 + 8\epsilon)B$.*

Our dynamic programming algorithm will find a solution with perceived cost at most $(1 + 5\epsilon)B$ and at most $3/\epsilon$ large arms per star, so the actual cost will be at most $(1 + 8\epsilon)B$.

3.2.2 Dynamic Programming

Before we formally define the subproblems of dynamic programming, we discuss the structure of a well-formed solution, say $\mathcal{S}$. We call a client covered by a small (large) arm a *small client* (*large client*), respectively. Let the *small span or s-span* of a star be the interval, possibly empty, formed from the left most to the right most small client in this star. Since the small arms do not intersect in $\mathcal{S}$, for any two s-spans I_1 and I_2 of two stars, either $I_1 \cap I_2 = \emptyset$ or $I_1 \subseteq I_2$ or $I_2 \subseteq I_1$. Therefore, the $\subseteq$ relation between s-span of stars in $\mathcal{S}$ defines a laminar family (a forest like structure).

Also, consider the restriction of $\mathcal{S}$ to the interval $I_{i,j}$ for two arbitrary points v_i and v_j. Assume that an arm has a direction and goes from the center of the star (i.e. the facility) to the client that it covers. There are some large arms that enter or leave this interval from v_i or v_j. There are two types of arms: the arms that enter the interval $I_{i,j}$ from v_i or v_j or the arms that leave the interval $I_{i,j}$ from v_i or v_j; for example a center inside the interval

$I_{i,j}$ might cover a client outside this interval or a center to the left of i might cover a client inside the interval or a client to the right of j. Note that a large arm may have both these types, i.e., it enters from one endpoint and leaves from the other endpoint. The arms that enter the interval can cover the *deficiency* of coverage for some client in the interval and the arms that leave the interval provide coverage for some client outside of interval and can be viewed as *surplus* to the demand of coverage of the clients in the interval. Also, recall that in the perceived cost, the length of a large arm is measured as an integer multiple q of $\epsilon^2 B$, where $0 \leq q \leq \frac{1}{\epsilon^2}$.

By the above observations, we can keep the information of all large arms that enter or exist the interval $I_{i,j}$ in four size $\frac{1}{\epsilon^2}+1$ vectors $\mathbf{D_i}, \mathbf{S_i}, \mathbf{D_j}$, and $\mathbf{S_j}$, which are the deficiency and surplus vectors of v_i and the deficiency and surplus vectors of v_j with respect to $I_{i,j}$, respectively. The qth entry ($0 \leq q \leq \frac{1}{\epsilon^2}$) of each of these vectors is an integer between 0 and $|\mathcal{C}|$. The qth element of vector $\mathbf{D_i}$ is the number of large arms entering from v_i having perceived length $q \cdot \epsilon^2 B$ *past* v_i. More specifically, it is the number of clients j' with $v_{j'} \geq v_i$ that are assigned to a center c with $v_c < v_i$ such that the interval $(v_i, v_{j'}]$ contains q multiples of $\epsilon^2 B$. In the same vein, entry q of $\mathbf{S_i}$ records the number of clients j' with $v_{j'} < v_i$ that are assigned to a center c with $v_c \geq v_i$ such that the interval $(v_c, v_i]$ contains q multiples of $\epsilon^2 B$. Similarly, $\mathbf{D}_j(q)$ is the number of large arms entering from v_j with perceived length q *prior to* v_j and $\mathbf{S}_i(q)$ is the number of large arms exiting from v_j with perceived length q past i.

3.3 The Table

The table we build in our dynamic programming algorithm captures "snapshots" of solutions bound between two given points plus some information on how arms cross these points. We consider the values $A(k', i, j, c, \beta, \mathbf{D_i}, \mathbf{D_j}, \mathbf{S_i}, \mathbf{S_j})$ corresponding to subproblems. The meanings of the parameters are as follows. 1) $1 \leq i \leq j \leq n$ corresponds to the interval $I_{i,j}$, 2) $0 \leq k' \leq k$ is the number of centers (of stars) in the interval $I_{i,j}$, 3) $c \in \mathcal{F}$ denotes a single point with either $c < i$ or $c > j$ (i.e. outside of $I_{i,j}$) that is the center of some star, or else $c = \perp$. If $c \neq \perp$ it is the only center outside of $I_{i,j}$ with small arms going into $I_{i,j}$ and the total cost of small arms that c pays to cover vertices in $I_{i,j}$ is β where $0 \leq \beta \leq (1+5\epsilon)B$ is an integer. 4) $\mathbf{D_i}, \mathbf{D_j}, \mathbf{S_i}, \mathbf{S_j}$ are deficiency and surplus vectors for the endpoints of interval $I_{i,j}$. Note that in the above, if $c = \perp$ then the value of β can be assumed to be zero. Let q denote the number of multiples of $\epsilon^2 B$ lying in the interval $(v_i, v_j]$.

The subproblem $A(k', i, j, c, \beta, \mathbf{D_i}, \mathbf{D_j}, \mathbf{S_i}, \mathbf{S_j})$ is true if and only if the following holds. It is possible to open k' centers in the interval $I_{i,j}$ and assign each $i' \in \mathcal{C}$ with $i \leq i' \leq j$: 1) to one of these open centers, or 2) to center c, if $c \neq \perp$, 3) or as a large arm exiting $I_{i,j}$, and also assign some of the large arms entering $I_{i,j}$ to these open centers such that: 1) the perceived load of each of the k' centers is at most $(1+5\epsilon)B$, 2) the load of c from small arms originating $i' \in \mathcal{C}$ with $i \leq i' \leq j$ is at most β, 3) and the large arms entering and/or exiting $I_{i,j}$ are *consistent* with $\mathbf{D_i}, \mathbf{D_j}, \mathbf{S_i}, \mathbf{S_j}$.

By consistent, we mean the following. First, for each $0 \leq a \leq \epsilon^{-2}$, each of the $\mathbf{D_i}(a)$ large arms entering $I_{i,j}$ is assigned to an open center f such that $(v_i, v_f]$ contains precisely a integer multiples of $\epsilon^2 B$, or (if $q \leq a$) exits $I_{i,j}$. A similar statement applies to $\mathbf{D_j}(a)$. Then for each $0 \leq a \leq \epsilon^{-2}$ we have that $\mathbf{S_j}(a)$ is precisely the number of large arms represented by $\mathbf{D_i}(a-q)$ that are not assigned to one of the k' open centers in the interval plus the number of large arms originating from clients in the interval that exit by passing v_j and have perceived length a past v_j. Finally, we also require that no open center serves more than $3/\epsilon$ clients using large arms.

The number of table entries is polynomial, because k', i, j, c are in $O(n)$ and β' is a

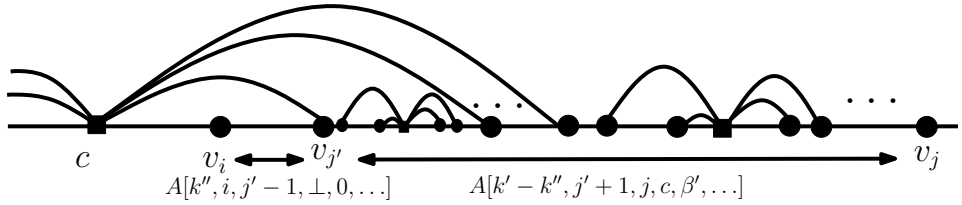


Figure 1 Case 1 of recursive step.

polynomial in n and $\frac{1}{\epsilon}$ and the deficiency and surplus vectors are in $O(n^{1+1/\epsilon^2})$, which is polynomial for a constant ϵ. We shortly explain how one can compute the table entries in polynomial time. After that, to find out if there is a feasible solution having perceived cost $(1+5\epsilon)B$, one simply needs to look at the value of $A[k, 1, n, \perp, 0, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}]$, where $\mathbf{0}$ is a vector having $1 + 1/\epsilon^2$ zero components.

3.4 The Recurrence

In this subsection we present the recurrence for the dynamic program of the PTAS for line.

Base Case. The base case is when $k' = 0$ and $i = j$. Without loss of generality assume that i is a client (or else there is nothing to be covered). First, assume $c = \perp$ and so $\beta = 0$ or $c \neq \perp$ but $\beta = 0$. In either case, v_i must be covered with a large arm. Assume this arm comes from left. In this case, the first component of $\mathbf{D_i}$, which corresponds to the number of large arms having perceived length $0 \cdot \epsilon^2 B = 0$ passed v_i, must be non-zero and one more than the first component of $\mathbf{S_j}$, because one will be used to cover i. All other components of these vectors must be the same. Also, all components of $\mathbf{S_i}$ and $\mathbf{D_j}$ must be the same. The case that the arm comes from right is similar. Now, assume $c \neq \perp$ and $\beta \neq 0$. Then, v_i is a small client and it must be covered by v_c. Therefore, $d(v_i, v_c)$ must be equal to β. Also, we must have $\mathbf{D_i} = \mathbf{S_j}$ and $\mathbf{S_i} = \mathbf{D_j}$. In all other cases, the entry of the table will be set to False.

Recursive Step. Next, we show how to determine if $A[k', i, j, c, \beta, \mathbf{D_i}, \mathbf{D_j}, \mathbf{S_i}, \mathbf{S_j}]$ is true when the parameters do not represent a base case by relating its value to values of smaller problems. In what follows, by *guessing* a parameter, we mean that we try all *polynomially* many possible values of that parameter and if one of them results in a feasible solution, we set the value of the current subproblem to true. We consider two cases regarding value of c:

1. $c \neq \perp$ and $\beta > 0$. There must be a small client in $I_{i,j}$ covered by c. We guess j' to be the leftmost small client in $I_{i,j}$ covered by c. Now, we can break the subproblem into two smaller subproblems at the left and right sides of j' (see Figure 1). If $j' = i$ or $j' = j$, one of the subproblems is empty and its value can be considered as true. Thus, assume $i < j' < j$. We guess k'' the number of stars having center in $I_{i,j'-1}$ and so the remaining $k' - k''$ centers will be in $I_{j'+1,j}$. Also, we guess the deficiency and surplus vectors, $\mathbf{D_{j'-1}}, \mathbf{S_{j'-1}}$, at $v_{j'-1}$ for the interval $I_{i,j'-1}$, and $\mathbf{D_{j'+1}}, \mathbf{S_{j'+1}}$, at $v_{j'+1}$ for the interval $I_{j'+1,j}$, such that all are consistent in the sense that $\mathbf{S_{j'-1}}(q) = \mathbf{D_{j'+1}}(q - q')$ where q' is the number of integer multiples of $\epsilon^2 B$ in $(v_{j'-1}, v_{j'+1}]$, and similarly for large arms crossing $v_{j'}$ from right to left.
   We check to see if $A[k'', i, j'-1, \perp, 0, \mathbf{D_i}, \mathbf{S_i}, \mathbf{D_{j'-1}}, \mathbf{S_{j'-1}}]$ is true. If it is true, we examine the second subproblem. The coverage that c provides to the right of j' can be computed as β' where $\beta' = \beta - d(v_c, v_{j'})$. We let $c' = c$ if $\beta' > 0$, and $c' = \perp$ if $\beta' = 0$. If

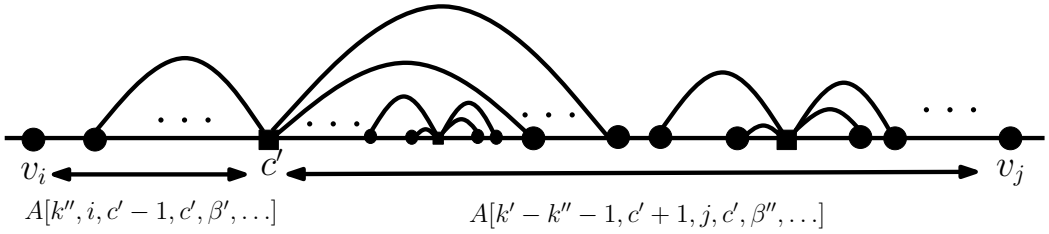


Figure 2 Case 2b of recursive step.

$A[k'-k'', j'+1, j, c, \beta', \mathbf{D_{j'+1}}, \mathbf{S_{j'+1}}, \mathbf{D_j}, \mathbf{S_j}]$ is also true, we set the value of subproblem true.

2. $c \neq \perp$ and $\beta = 0$, or $c = \perp$. We consider two subcases regarding value of k':
   - **(a)** $k' = 0$. All clients in $I_{i,j}$ must be covered by large arms from centers (facilities) outside the interval. First suppose that i is a client and, without loss of generality, assume v_i is covered by a large arm from the left. Then, the number of large arms having length $0 \cdot \epsilon^2 B = 0$ passed v_i must be non-zero and we use one such arm to cover v_i. In this case we define $\mathbf{D'_i} = \mathbf{D_i} - (1, 0, \ldots, 0)$, i.e., the updated deficiency vector after covering i. If i is not a client (and so does not need to be covered) we define $\mathbf{D'_i} = \mathbf{D_i}$. In both cases (whether i is a client or not) suppose that $(v_i, v_{i+1}]$ has q multiples of $\epsilon^2 B$ for some $0 \leq q \leq 1/\epsilon^2$. Thus, the value of the first q components in $\mathbf{D'_i}$ must be zero. Define a deficiency vector $\mathbf{D_{i+1}}$ for $i+1$, which is equal to the vector obtained by shifting the values of $\mathbf{D'_i}$, q places to the left (add trailing zeros for the values missing). Also, the last q components of $\mathbf{S_i}$ must be zero, too (or else the arms that start at a node $v_{j'} \geq v_{i+1}$ and exits v_i will have length larger than B). Define a surplus vector $\mathbf{S_{i+1}}$ for $i+1$, which is equal to the vector obtained by shifting the values of $\mathbf{S_i}$, q places to the right (add leading zeros for the values missing). We set the value of this subproblem to $A[0, i+1, j, \perp, 0, \mathbf{D_{i+1}}, \mathbf{S_{i+1}}, \mathbf{D_j}, \mathbf{S_j}]$.
   - **(b)** $k' > 0$. Note that since $c \neq \perp$ and $\beta = 0$, or $c = \perp$, no small arm can enter $I_{i,j}$. Consider the set of centers in $I_{i,j}$. The s-span (interval of small arms) of these centers forms a laminar family. Consider the roots of the forest of this laminar family and let c' be the center corresponding to the leftmost root; we guess c' (see Figure 2). Observe that the s-span of c' is not contained in the s-span of any other star having a center in $I_{i,j}$. This star has at most $3/\epsilon$ large arms. Recall that in the perceived cost of a star, the length of large arms is measured in multiples of $\epsilon^2 B$. For each $0 \leq q \leq 1/\epsilon^2$, we guess $n_q^{(l)}$ and $n_q^{(r)}$ the number of length $q \cdot \epsilon^2 B$ large arms that c' has (with respect to perceived cost) to its left and its right, respectively. We also guess k'' the number of stars having center in $I_{i,c'-1}$. We must have $k'-k''-1 \geq 0$ stars having center in $I_{c'+1,j}$. Also, we guess β' where c' provides β' coverage to its left side. Finally, we guess the deficiency and surplus vectors, $\mathbf{D_{c'-1}}, \mathbf{S_{c'-1}}$, at $v_{c'-1}$ for the interval $I_{i,c'-1}$ and we guess the deficiency and surplus vectors, $\mathbf{D_{c'+1}}, \mathbf{S_{c'+1}}$, at $v_{c'+1}$ for the interval $I_{c'+1,j}$ and make sure that these vectors are consistent in the sense that $\mathbf{D_{c'+1}}(q-q') = \mathbf{S_{c'-1}}(q) - n_q^{(l)}$ where q' is the number of integer multiples of $\epsilon^2 B$ in $(v_{c'-1}, v_{c'+1}]$, and similarly for the large arms crossing c' from right to left.
     Now, we can break the subproblem into two smaller subproblems at the left and right sides of c'. We first check to see if $A[k'', i, c'-1, c', \beta', \mathbf{D_i}, \mathbf{S_i}, \mathbf{D_{c'-1}}, \mathbf{S_{c'-1}}]$ is

true (if $\beta' = 0$, we check $A[k'', i, c' - 1, \perp, 0, \mathbf{D_i}, \mathbf{S_i}, \mathbf{D_{c'-1}}, \mathbf{S_{c'-1}}]$). We guess β'' such that $\beta' + \beta'' + \sum_{q=0}^{\frac{1}{\epsilon^2}}(n_q^{(l)} + n_q^{(r)})q$, where the cost of small arms of c' to clients in $I_{c'+1,j}$ is β''. We check and if $A[k' - k'' - 1, c' + 1, j, c', \beta'', \mathbf{D_{c'+1}}, \mathbf{S_{c'+1}}, \mathbf{D_j}, \mathbf{S_j}]$ is also true, we set the value of subproblem true (again if $\beta'' = 0$, we check $A[k' - k'' - 1, c' + 1, j, \perp, 0, \mathbf{D_{c'+1}}, \mathbf{S_{c'+1}}, \mathbf{D_j}, \mathbf{S_j}]$).

4 Tree Metrics

The extension of the PTAS presented for line metrics to tree metrics is not clear. However, there is a QPTAS for line metrics (see [10]) which uses a somewhat different approach. We describe the high level idea of that QPTAS (for line metrics) here and refer the reader to [10] for details. Then we explain how that approach can be extended to a QPTAS for tree metrics.

As before assume we have our points $v_1 \le v_2 \le \ldots \le v_n$ on a line and we have a guessed bound B on the value of optimum. With a similar scaling approach as in Section 3 we can assume that the minimum distance between two consecutive points $i, i+1$ is at least $\epsilon B/n^2$ and at most B; thus the maximum pairwise distance is at most nB. Scaling everything by $\epsilon B/n^2$ we can assume the minimum distance is at least 1 the maximum distance between consecutive points is n^3/ϵ and that B is at most n^4/ϵ. This will increase the cost of the solution to at most $(1+\epsilon)B$. We then use a dynamic programming (DP) that computes a $(1+\epsilon)$-approximate solution to an instance satisfying above conditions. Each subproblem is defined by an interval $I_{i,j}$ and parameter k', and the goal is to cover the clients in this interval with k' stars whose centers are in this interval and some other stars whose centers are outside. We use a binary dissection to break the problem into two (almost) equal parts $I_{i,m}$ and $I_{m+1,j}$ where m is the middle point. This gives rise to a dissection tree of height $O(\log n)$ with the interval $I_{1,n}$ at the root and n singleton intervals $I_{i,i}$ as leaves. So the height of the recursion is $O(\log n)$. As before we keep deficiencies and surpluses vectors for the two ends v_i and v_j: $\mathbf{D_i}, \mathbf{D_j}, \mathbf{S_i}, \mathbf{S_j}$, but they are defined slightly differently. Consider vector $\mathbf{S_i} = (s_1^{(i)}, \ldots, s_\sigma^{(i)})$ (with σ to be defined soon). Each s_a will keep the number of clients to the left of this interval (i.e. before v_i) that are at an approximate distance l_a and are served by centers to the right of i (possibly after j). Similarly $\mathbf{S_j}$ stores the number of clients to the right of v_j that are served by centers before j. $\mathbf{D_i}$ and $\mathbf{D_j}$ will be representing the number of clients inside $I_{i,j}$ within an approximate certain distance from v_i (or v_j) that are to be covered by a center outside of the interval. To cut down on the interface of an interval $I_{i,j}$ with the rest of the line, we round up the surplus and deficiency lengths on each of the right and left sides to the nearest power of $(1+\epsilon'/\log n)$, for some ϵ' depending on ϵ, at each level of dissection. Thus, we only keep track of lengths $l_a = (1+\epsilon'/\log n)^a$, $a \in \{1, \ldots, \sigma\}$. For instance, for $\mathbf{S_i} = (s_1^{(i)}, \ldots, s_\sigma^{(i)})$, $s_a^{(i)}$ will be the number of clients to the left of $I_{i,j}$ that are at (scaled up) distance $(1+\epsilon'/\log n)^a$ from i that are served by centers inside the interval $I_{i,j}$. So there will be $\sigma = O(\log n \cdot \log B/\epsilon') = O(\log^2 n/\epsilon')$ different lengths and as a result at most $n^{O(\log^2 n/\epsilon')}$ different surplus and deficiency vectors. In this way, each arm of a star will be scaled up by a factor of at most $(1+\epsilon'/\log n)$ at each level of DP computation (to account for the rounding), and since the depth of recursion (dissection) is $\lceil \log n \rceil$, this will result in an extra factor of $(1+\epsilon'/\log n)^{\lceil \log n \rceil} \le (1+\epsilon)$ (for a suitable choice of ϵ') over the entire length of each arm. In other words, if a subproblem for an interval i, j and parameter k' is feasible (with each star costing at most B) without rounding the lengths of deficiency and surplus vectors then the subproblem with rounded (up to nearest power of $(1+\epsilon'/\log n)$) lengths for deficiency and surplus vectors is feasible if each star is allowed to have cost at most $(1+\epsilon) \cdot B$.

Each entry of the table represents a subproblem $(i, j, k', \mathbf{D_i}, \mathbf{D_j}, \mathbf{S_i}, \mathbf{S_j})$, where:

1. i, j represents the interval $I_{i,j}$.
2. k' is the number of centers to be opened from among the points in $I_{i,j}$.
3. $\mathbf{D_i} = (d_1^{(i)}, \ldots, d_\sigma^{(i)})$ and $\mathbf{D_j} = (d_1^{(j)}, \ldots, d_\sigma^{(j)})$ are the deficiency vectors on the left and right sides of the interval $I_{i,j}$, respectively.
4. $\mathbf{S_i} = (s_1^{(i)}, \ldots, s_\sigma^{(i)})$ and $\mathbf{S_j} = (s_1^{(j)}, \ldots, s_\sigma^{(j)})$ are the surplus vectors on the left and right sides of $I_{i,j}$, respectively.

Each surplus and deficiency vector is a vector of size $\sigma = O(\log^2 n/\epsilon')$, where $d_a^{(p)}$ or $s_a^{(p)}$ (for $p \in \{i, j\}$) is the number of broken arm parts of length $(1 + \epsilon'/\log n)^a$ (after rounding). Each entry of the table records in Boolean values the feasibility of having k' stars centered in the points in $I_{i,j}$, such that each star has cost at most $(1 + \epsilon) \cdot B$. Each of the k' stars would cover some clients in $I_{i,j}$ and the clients located at distances $\mathbf{S_i}$ and $\mathbf{S_j}$ from the endpoints i and j of the interval. The rest of the clients have to be covered with the broken arms of $\mathbf{D_i}$ and $\mathbf{D_j}$, thus connected to the two sides i and j, respectively. The size of the DP table is $O(n^2 \cdot k \cdot n^{O(\log n \log B/\epsilon')}) = n^{O(\log^2 n/\epsilon')}$, which is quasi-polynomial in n. See [10] for the details of how to fill in the entries of this table.

Now suppose that the given metric for the MLkFL instance can be represented as a cost function on the edges of a tree T. The algorithm, as before, works with a guessed value B as an upper bound for L^{opt}. Also, using a scaling argument as for the case of line metrics, we can assume that the aspect ratio of heaviest to cheapest edge cost is polynomially bounded. Next, we can make the tree T binary by introducing zero-cost edges at nodes that have more than two children, keeping one of its children and placing the rest as a subtree hanging from the zero-cost edge added. Repeating this gives a binary tree that still has linear size. So for the rest of this section we assume that the input tree is binary.

For each binary tree with n nodes one can find an edge $e = (u, v)$ (where u is parent of v) such that each subtree resulted by deleting e has size in $[n/3, 2n/3]$. This splitting of the tree into two subtrees T_v (tree rooted at v) and $T \setminus T_v$ that are almost the same size (by a factor of at most two) plays the role breaking the problem into two almost equal sizes. Given a binary tree T we can recursively partition it into two "almost equal" subtrees until we arrive at subtrees of size 1. The depth of this recursive dissection will be $O(\log n)$ and each time we recursively break the tree into two smaller binary trees (whose sizes differ by a factor of at most 2). The breaking point introduces a new interface (or "portal") point for the two smaller sub-trees: if edge $e = (u, v)$ is cut then v is an interface point (or portal) for the subproblem T_v in addition to any other interface point it might have had passed on to from previous dissection operations, and u is an interface point (portal) for $T \setminus T_v$ in addition to any other portal points generated before. More specifically, each subproblem is of the form $(T', k', \{S^{(p)}\}_{p \in P(T')}, \{D^{(p)}\}_{p \in P(T')})$ where T' is a subtree that is obtained by performing the dissection operation, $0 \leq k' \leq k$ is the number of centers of stars to be opened in T', $P(T') \subseteq V(T')$ is the set of portal points of T'. If a tree $\hat{T}$ is cut into two almost equal sized subtrees T_1 (rooted at v) and $T_2 = \hat{T} \setminus T_1$ by cutting edge $e = (u, v)$ then $P(T_1)$ will consist of all the portals of $\hat{T}$ that are in T_1 plus node v. Similarly $P(T_2)$ consists of all the portals of $\hat{T}$ that are in T_2 plus node u. It follows that for each subproblem, the number of portals is at most $O(\log n)$. The dynamic programming then follows along the same lines as the QPTAS described above (see [10] for details) for the line metrics, and we obtain the following theorem.

▶ **Theorem 9.** *For any constant $0 < \varepsilon \leq 1$, there is a $(1 + \varepsilon)$-approximation algorithm for* MLkFL *on tree metrics that runs in quasi-polynomial time.*

4.1 Star Metrics

We now consider MLkFL in star metrics, but in the more-general setting where each client j has an integer demand D_j that may be split integrally across various open facilities; we call this an *integer-splittable assignment*. The load of a facility i is now defined as $\sum_j x_{ij} d(i,j)$ where $x_{ij} \in \mathbb{Z}_{\geq 0}$ is the amount of j's demand that is served by i. We devise a 14-approximation algorithm for this problem. At a high level our approach is similar to the one used to obtain the PTAS for line metrics. We again "guess" the optimal value B. We argue via a slightly different uncrossing technique that if $B \geq L^{opt}$, then there exists a well-structured fractional solution with maximum load at most $6B$, and use DP to obtain a fractional solution with maximum load at most $12B$. This can then be converted to an integer-splittable assignment with maximum load at most $14B$ using the GAP-rounding algorithm, since it is easy to ensure via some preprocessing that $d(i,j) \leq 2B$ for every facility i and client j. Thus, we either determine that $B < L^{opt}$ or obtain a solution with maximum load at most $14B$. The details are deferred to the full version of this paper.

▶ **Theorem 10.** *There is a 14-approximation algorithm for* MLkFL *on star metrics with non-uniform demands and integer-splittable assignments.*

5 Hardness Results and Integrality-gap Lower Bounds

We now present various hardness and integrality-gap results. We prove that MLkFL is strongly *NP*-hard on line metrics and *APX*-hard in the Euclidean plane (Theorems 11 and 12). We also demonstrate that a natural configuration-style LP has an unbounded integrality gap (Theorem 13). The details of the first two theorems are deferred to the full version of this paper.

▶ **Theorem 11.** *Minimum-load k-facility location is strongly NP-hard even in line metrics.*

▶ **Theorem 12.** *It is NP-hard to α-approximate minimum-load k-facility location problem on the Euclidean plane, for any $\alpha < 4/3$. Thus,* MLkFL *is* APX*-hard in the Euclidean plane.*

5.1 Integrality-gap Lower Bound

Let $(\mathcal{F}, \mathcal{C}, d, k)$ be an MLkFL instance. Given a candidate "guess" B of the optimal value, we can consider the following LP-relaxation of the problem of determining if there is a solution with maximum load at most B. We propose the following linear programming for the MLkFL . For each facility $i \in \mathcal{F}$, define $\mathcal{S}(B;i) := \{C \subseteq \mathcal{C} : \sum_{j \in C} d(i,j) \leq B\}$ to be the set of all stars centered at i that induce load at most B at i. We will often refer to a star in $\mathcal{S}(B;i)$ as a configuration. (Note that $\mathcal{S}(B;i)$ contains $\emptyset$.) Our LP Will be a *configuration-style LP*, where for every facility i and star $C \in \mathcal{S}(B;i)$, we have a variable denoting if star C is chosen for facility i. This yields the following natural feasibility LP.

$$\left.\begin{aligned}
\sum_{i \in \mathcal{F}} \sum_{C \in \mathcal{S}(B;i): j \in C} x(i,C) &\geq 1 \qquad \forall j \in \mathcal{C} && (1)\\
\sum_{C \in \mathcal{S}(B;i)} x(i,C) &\leq 1 \qquad \forall i \in \mathcal{F} && (2)\\
\sum_{i \in \mathcal{F}} \sum_{C \in \mathcal{S}(B;i)} x(i,C) &\leq k && (3)\\
x &\geq 0.
\end{aligned}\right\} \quad \text{(P)}$$

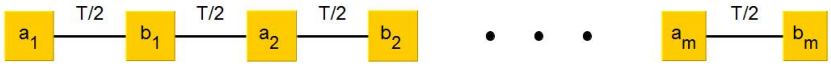


Figure 3 Example showing bad integrality gap for the configuration LP in line metric.

Constraint (1) ensures that each client belongs to some configuration, and constraints (2) and (3) ensure that each facility belongs to at most one configuration, and that there are at most k configurations. We show that there is an MLkFL instance on the line metric, where the smallest value B_{LP} for which (P) is feasible is smaller than the optimal value by an $\Omega(k/\log k)$ factor; thus, the "integrality gap" of (P) is $\Omega(k/\log k)$. Moreover, in this instance, the graph containing the (i,j) edges such that $d(i,j) \leq B_{\mathrm{LP}}$ is connected.

▶ **Theorem 13.** *The integrality gap of* (P) *is* $\Omega(k/\log k)$ *even for line metrics.*

Proof. Assume for simplicity that k is odd (the argument easily extends to even k). Consider the following simple MLkFL instance. We have $\mathcal{F} = \{a_1, b_1, a_2, b_2, \ldots, a_m, b_m\}$, where $2m = k+1$. These facilities are located on a line as shown in Figure 3, with the distance between any two consecutive nodes being $T/2$. There are $n = 2k$ clients colocated with each facility. Let A_i (respectively B_i) denote the set of clients located at a_i (respectively b_i) for $1 \leq i \leq m$.

There is a feasible solution to (P) with $B = T$. For all $i = 1, \ldots, m$, we set $x(a_i, A_i \cup \{j, j'\}) = \frac{k}{(k+1)\cdot\binom{n}{2}}$ for all $j, j' \in B_i$; note that all these configurations lie in $\mathcal{S}(T; a_i)$. Similarly, we set $x(b_i, B_i \cup \{j, j'\}) = \frac{k}{k+1\cdot\binom{n}{2}}$ for all $j, j' \in A_i$. It is easy to verify that x is a feasible solution. It is clear that constraints (2) and (3) hold since every facility belongs to exactly $\binom{n}{2}$ configurations. Consider a client $j \in A_i$. j is covered to an extent of $\frac{k}{k+1}$ by the $\binom{n}{2}$ configurations $\left\{A_i \cup \{k, \ell\}\right\}_{k,\ell \in B_i}$ in $\mathcal{S}(a_i; T)$ and to an extent of $\frac{1}{k+1}$ by the $n-1$ configurations $\left\{B_i \cup \{j, k\}\right\}_{k \in A_i : k \neq j}$. A symmetric argument applies to clients in some B_i set. (If k is even, we may set $B = 2T$ and choose the above configurations for the first $k-2$ facilities and the k-th facility; for facility $k-1$, we consider $\binom{n}{2}$ configurations, each of which contains all the clients colocated at facility $k-1$, two clients colocated with the $(k-2)$-th facility and 2 clients colocated with the k-th facility.)

Finally, we show that any feasible solution must have maximum load at least $T \cdot \frac{k}{2H_k}$, where $H_r := 1 + \frac{1}{2} + \ldots + \frac{1}{r}$ is the r-th harmonic number, which proves the theorem. In any feasible solution, there is some location v that does not have an open facility. For $i = 1, \ldots, k$, let n_i be the number of clients colocated at v that are assigned to a facility at a location that is i hops away from v; set $n_i = 0$ if there is no such location. Then, $\sum_{i=1}^{k} n_i = n$, and the maximum load L at a facility is at least $\max_{i=1,\ldots,k} \frac{n_i i T}{4}$ since there are at most two facilities that are i hops away from v, and one of them must have at least $\frac{n_i}{2}$ clients assigned to it. Thus, we have $n_i \leq \frac{4L}{iT}$ for all $i = 1, \ldots k$, and so $n \leq \frac{4L}{T} \cdot H_k$, or $L \geq \frac{nT}{4H_k}$. (Note that this argument does not depend on whether k is odd or even.) ◀

6 Concluding Remarks

In this paper we present the first true polynomial time approximation for the MLkFL restricted to line metrics and the first true approximation in tree metrics. We also show that the standard tools of LP rounding (even for configuration based LP) or local search methods, which have been used successfully for various facility location problems do not seem to work

for this problem (even for this restricted metrics). Obviously, the major open question here is to obtain a true approximation (even an $O(\log n)$-approximation) for the MLkFL on general metrics. A smaller step could be to obtain such an algorithm for the Euclidean metrics. Note that the *APX*-hardness result for the Euclidean metrics shows that this problem is significantly more difficult than the uncapacitated facility location or k-median (for which there are known PTAS's).

References

1. H.-C. An, A. Bhaskara, C. Chekuri, S. Gupta, V. Madan, and O. Svensson. Centrality of trees for capacitated k-center. In *Proceedings of APPROX*, 2014.
2. E.M. Arkin, R. Hassin, and A. Levin. Approximations for minimum and min-max vehicle routing problems. *Journal of Algorithms*, 59(1):1–18, 2006.
3. V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004.
4. M. Charikar, S. Guha, É. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k-median problem. *Journal of Computer and System Sciences*, 65(1):129–149, 2002.
5. M. Cygan, M. T. Hajiaghayi, and S. Khuller. Lp rounding for k-centers with non-uniform hard capacities. *Arxiv preprint arXiv:1208.3054*, 2012.
6. G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha. Covering graphs using trees and stars. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 24–35, 2003.
7. D. Hochbaum and D. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: using the dual approximation approach. *SIAM Journal on Computing*, 17:539–551, 1988.
8. K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani. Greedy facility location algorithms analyzed using dual-fitting with factor-revealing lp. *Journal of the ACM*, 50(6):795–824, 2003.
9. K. Jain and V. V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *Journal of the ACM*, 48(2):274–296, 2001.
10. A. Jorati. Approximation algorithms for min-max vehicle routing problems. Master's thesis, University of Alberta, Department of Computing Science, 2013.
11. M. R. Khani and M. R. Salavatipour. Improved approximation algorithms for the min-max tree cover and bounded tree cover problems. In *APPROX*, 2011.
12. S. Li and O. Svensson. Approximating k-median via pseudo-approximation. In *Symposium on Theory of Computing (STOC)*, 2013.
13. P. Mirchandani and R. Francis, editors. *Discrete location theory*. Jown Wiley and Sons, 1990.
14. H. Nagamochi and K. Okada. Approximating the minmax rooted-tree cover in a tree. *Information Processing Letters*, 104(5):173–178, 2007.
15. R. Ravi. Workshop on Flexible Network Design, 2012. `http://fnd2012.mimuw.edu.pl/qa/index.php?qa=4&qa_1=approximating-star-cover-problems`.
16. D. B. Shmoys. The design and analysis of approximation algorithms: facility location as a case study. In S. Hosten, J. Lee, and R. Thomas, editors, *Trends in Optimization, AMS Proceedings of Symposia in Applied Mathematics 61*, pages 85–97. 2004.
17. D. B. Shmoys and E. Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62(3):461–474, 1993.

The Cover Number of a Matrix and its Algorithmic Applications

Noga Alon[1,2], Troy Lee[3], and Adi Shraibman[4]

1 **Sackler School of Mathematics and Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel**
nogaa@tau.ac.il
2 **Institute for Advanced Study, Princeton, New Jersey, 08540, USA**
3 **School of Physics and Mathematical Sciences, Nanyang Technological University and Centre for Quantum Technologies, Singapore**
troyjlee@gmail.com
4 **School of Computer Science. Academic College of Tel-Aviv Yaffo, Israel**
adish@mta.ac.il

Abstract

Given a matrix A, we study how many ϵ-cubes are required to cover the convex hull of the columns of A. We show bounds on this cover number in terms of VC dimension and the γ_2 norm and give algorithms for enumerating elements of a cover. This leads to algorithms for computing approximate Nash equilibria that unify and extend several previous results in the literature. Moreover, our approximation algorithms can be applied quite generally to a family of quadratic optimization problems that also includes finding the densest k-by-k combinatorial rectangle of a matrix. In particular, for this problem we give the first quasi-polynomial time additive approximation algorithm that works for any matrix $A \in [0,1]^{m \times n}$.

1998 ACM Subject Classification G.1.2 Approximation

Keywords and phrases Approximation algorithms, Approximate Nash equilibria, Cover number, VC dimension

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.34

1 Introduction

Consider a quadratic optimization problem where we wish to maximize $p^T Aq$ over probability distributions p, q, subject to linear constraints. Examples of problems of this type include Nash equilibrium and the densest combinatorial rectangle problem. A general scheme for finding an approximately optimal solution is based on the following notion of an ϵ-net for an m-by-n matrix A. Denote by $\mathrm{conv}(A)$ the convex hull of the columns of A. We call a set of vectors $S \subseteq \mathbb{R}^m$ an ϵ-net for A if for all $v \in \mathrm{conv}(A)$ there is a vector $u \in S$ such that $\|v - u\|_\infty \leq \epsilon$. An efficient means to enumerate elements of an ϵ-net S for A gives an efficient means for finding a near optimal solution to the original quadratic optimization problem: for each $u \in S$ solve the linear program to maximize $p^T u$ over probability distributions p, q, subject to the original linear constraints on p and q and the additional constraint $\|u - Aq\|_\infty \leq \epsilon$. The largest such value will be within 2ϵ of the optimal and the running time of this approximation algorithm will be a polynomial factor times the time required to enumerate an ϵ-net for A. This approximation algorithm motivates the study of ϵ-nets and efficient algorithms for enumerating them.

Say that $A \in [-1,1]^{m \times n}$. Denote by $N_\epsilon(A)$ the minimal size of an ϵ-net for A, which we will also informally refer to as the *cover number* of A. An obvious upper bound on $N_\epsilon(A)$ is

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 34–47

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

$(1/\epsilon)^m$. This naive bound can be improved by realizing that the convex hull of the columns of A actually lives in a space of dimension $\mathrm{rank}(A)$, which allows an improvement to $N_\epsilon(A) = [O(1/\epsilon)]^{\mathrm{rank}(A)}$. Recently, [7] made this bound algorithmic, showing that an ϵ-net for A can be enumerated by a randomized Las Vegas algorithm in time $[O(1/\epsilon)]^{\mathrm{rank}(A)}\mathrm{poly}(mn)$. Following the above approximation paradigm, this led to polynomial time additive approximation schemes for two-player Nash Equilibrium when the sum of the payoff matrices has logarithmic rank, improving work of Kannan and Theobold [17] who showed the same when the sum of the payoff matrices has constant rank. The [7] bound on the cover number combined with the above approximation paradigm also gave an efficient approximation algorithm for finding the densest k-by-k combinatorial rectangle provided the associated matrix has rank at most logarithmic in the dimension.

In this paper, we continue the study of $N_\epsilon(A)$ and its relation to other complexity measures of A, like VC dimension, γ_2 norm, and communication complexity (these measures are formally defined in the sequel). In particular, we show that $N_\epsilon(A) = n^{O(\mathrm{VC}(A)/\epsilon^2)}$ and that an ϵ-net can be enumerated deterministically in the same time. As $\mathrm{VC}(A) \leq \log(m)$ for any matrix with m rows, this recovers the quasi-polynomial time approximation for Nash equilibrium shown by Lipton et al. [19], and also gives a quasi-polynomial time additive approximation algorithm for the densest k-by-k combinatorial rectangle problem.

By the triangle inequality it is easy to see that an $\epsilon/2$-net for a matrix B satisfying $\|A - B\|_\infty \leq \epsilon/2$ gives an ϵ-net for A (here $\|X\|_\infty$ denotes the largest absolute value of an entry of X). Thus to construct ϵ-nets for A, it suffices to look for "simpler" matrices that are entrywise close to A. Define the ϵ-approximate rank of A as $\mathrm{rank}_\epsilon(A) = \min_{B:\|A-B\|_\infty \leq \epsilon} \mathrm{rank}(B)$. Existentially $N_\epsilon(A) \leq [O(1/\epsilon)]^{\mathrm{rank}_{\epsilon/2}(A)}$, but for the algorithm of [7] to enumerate such a cover, it explicitly needs to find an approximating matrix B whose rank is equal to $\mathrm{rank}_{\epsilon/2}(A)$. We currently do not know an algorithm to do this working in time $[O(1/\epsilon)]^{\mathrm{rank}_{\epsilon/2}(A)}$, or even $(n/\epsilon)^{\mathrm{rank}_{\epsilon/2}(A)}$ for that matter.

For a sign matrix A and any $\epsilon < 1$, it is easy and known that $\mathrm{VC}(A) \leq \mathrm{rank}_\epsilon(A)$. Thus the results in this paper give a way to enumerate an ϵ-net for a sign matrix A in deterministic time $n^{O(\mathrm{rank}_\epsilon(A)/\epsilon^2)}$. We present a similar result in terms of the γ_2 norm. The γ_2 norm, also known as the Hadamard product operator norm, has recently seen many applications in communication complexity and learning theory [20, 24, 25, 23]. Part of its usefulness is that the approximate γ_2 norm serves as a proxy for the approximate rank and can be computed efficiently via semidefinite programming. Based on the γ_2 norm, we show a Las Vegas randomized algorithm for enumerating an ϵ-net for A in time $(1/\epsilon)^{r\log(r)\log(mn)/\epsilon^2}$ where $r = \mathrm{rank}_{\epsilon/4}(A)$. While being a slightly weaker result than the one using the VC dimension, this has the benefit of having a simple self-contained proof.

2 Algorithmic Applications

We first show how efficient constructions of an ϵ-net for A lead to approximation algorithms for Nash equilibria and finding a densest combinatorial rectangle. This idea was already presented in [7] for ϵ-nets constructed from low rank decompositions of A. We present the proof again here in a slightly more general form for completeness.

2.1 Approximate Nash Equilibria

Let $A, B \in [-1, 1]^{m\times n}$ be the payoff matrices of the row and column players of a 2-player game. In other words, $A(i, j)$ is the payoff to Alice when she plays strategy i and Bob plays strategy j, and similarly $B(i, j)$ is the payoff to Bob when Alice plays strategy i and he plays

1. Create an $\epsilon/2$-net S for $A+B$. For each $u \in S$, solve the following linear program:
$$\begin{array}{ll} \underset{p\in\Delta_m, q\in\Delta_n}{\text{maximize}} & p^T u - \max_i e_i^T Aq - \max_j p^T Be_j \\ \text{subject to} & \|(A+B)q - u\|_\infty \leq \epsilon/2. \end{array}$$
2. Output p, q that achieve an objective value at least $-\epsilon$.

Figure 1 Finding ϵ-Nash equilibrium for payoff matrices A, B given an $\epsilon/2$-net for $A+B$.

strategy j. Let $\Delta_n = \{p \in \mathbb{R}^n : \|p\|_1 = 1, p \geq 0\}$ be the set of n-dimensional probability vectors. A Nash equilibrium is a pair of strategies (p, q) for $p \in \Delta_m, q \in \Delta_n$ satisfying

$$\begin{aligned} p^T Aq &\geq e_i^T Aq \quad \forall i \in \{1, \ldots, m\} \\ p^T Bq &\geq p^T Be_j \quad \forall j \in \{1, \ldots, n\} \end{aligned}$$

Here e_i denotes the vector with a 1 in the ith position and zeros elsewhere.

Alternatively, a Nash equilibrium is a solution to the following optimization problem:

$$\max_{p\in\Delta_m, q\in\Delta_n} p^T(A+B)q - \max_i e_i^T Aq - \max_j p^T Be_j \tag{1}$$

An ϵ-Nash equilibrium is a pair of strategies with the property that each player's payoff cannot improve by more than ϵ by moving to a different strategy, i.e.,

$$\begin{aligned} p^T Aq &\geq e_i^T Aq - \epsilon \quad \forall i \in \{1, \ldots, m\} \\ p^T Bq &\geq p^T Be_j - \epsilon \quad \forall j \in \{1, \ldots, n\} \end{aligned}$$

▶ **Lemma 1.** *Any $p \in \Delta_m, q \in \Delta_n$ that achieve an objective value at least $-\epsilon$ for (1) form an ϵ-Nash equilibrium.*

We now show how to find an ϵ-Nash equilibrium for a game with payoff matrices A, B given an ϵ-net for $A+B$. The algorithm is described in Figure 1.

▶ **Theorem 2.** *Let $A, B \in [-1, +1]^{m\times n}$. Suppose there is a deterministic (or Las Vegas randomized) algorithm running in time t for enumerating an $\epsilon/2$-net for $A+B$. Then an ϵ-Nash equilibrium for the game with payoff matrices A, B can be found by a deterministic (or Las Vegas randomized) algorithm in time $t \cdot \text{poly}(mn)$.*

Proof. The algorithm to find an ϵ-Nash equilibrium enumerates all vectors u in an $(\epsilon/2)$-net for $A+B$. For each of these vectors the algorithm solves the following program:

$$\begin{array}{ll} \underset{p\in\Delta_m, q\in\Delta_n}{\text{maximize}} & p^T u - \max_i e_i^T Aq - \max_j p^T Be_j \\ \text{subject to} & \|(A+B)q - u\|_\infty \leq \epsilon/2. \end{array}$$

Let $p_* \in \Delta_m, q_* \in \Delta_n$ be a Nash equilibrium, and so $0 = p_*^T(A+B)q_* - \max_i e_i^T Aq_* - \max_j p_*^T Be_j$. For u in the $\epsilon/2$-net satisfying $\|(A+B)q_* - u\|_\infty \leq \epsilon/2$ we then have

$$\begin{aligned} \max_{p,q} p^T u - \max_i e_i^T Aq - \max_j p^T Be_j &\geq p_*^T u - \max_i e_i^T Aq_* - \max_j p_*^T Be_j \\ &\geq p_*^T(A+B)q_* - \max_i e_i^T Aq_* - \max_j p_*^T Be_j - \epsilon/2 \geq -\epsilon/2 \end{aligned}$$

Thus the algorithm finds a pair p, q such that the optimal value is at least $-\epsilon/2$. By going via the $\epsilon/2$-net again, and using Lemma 1 we see that p, q are an ϵ-Nash equilibrium. ◀

2.2 Densest Combinatorial Rectangle

For a matrix $A \in [0,1]^{m\times n}$ and subsets S, T of rows and columns, let $A_{S,T}$ be the submatrix induced by S and T. The density of the submatrix $A_{S,T}$ is

$$\text{density}(A_{S,T}) = \frac{\sum_{i\in S, j\in T} A_{ij}}{|S||T|},$$

that is, the average of the entries in $A_{S,T}$.

▶ **Definition 3** (Densest k-by-k combinatorial rectangle)**.** Let $A \in [0,1]^{m\times n}$. The densest k-by-k combinatorial rectangle problem is to find sets S_*, T_*, each of size k, such that

$$\text{density}(A_{S_*,T_*}) = \max_{S,T:|S|=|T|=k} \text{density}(A_{S,T})\,.$$

Sets S, T which achieve the maximum up to an additive ϵ we call an ϵ-approximate densest k-by-k combinatorial rectangle.

A closely related problem is the densest k-subgraph problem. Here the goal is to find a set S_* realizing $\max_{S,|S|=k} \text{density}(A_{S,S})$. This problem is NP-hard and Khot has also shown that it does not have a PTAS unless NP $\subseteq$ BPTIME(2^{n^ϵ}) [16]. The best known polynomial time algorithm guarantees an optimal solution within a multiplicative factor of $n^{1/4+\epsilon}$ of the optimal density [9]. For dense graphs (with at least an ϵ-fraction of edges), Arora, Karger, and Karpinski give a polynomial time approximation scheme for $k = \Omega(n)$ [4]. This also follows from the results in [2].

It is straightforward to see that the density of the densest k-by-k combinatorial rectangle and the densest $2k$-subgraph differ by at most a factor of 2. Thus hardness results for densest k-subgraph carry over to densest k-by-k combinatorial rectangle. We note that as shown in [1], assuming that the *Hidden Clique Problem*, that is, the problem of finding a planted clique of size $n^{1/3}$ in the random graph $\mathcal{G}(n, 1/2)$ is hard, then so is approximating the Densest k-Subgraph to within any constant factor or to within any additive error bounded away from 1, for a subgraph of size $k = n^{1-\epsilon}$ for any $2/3 \geq \epsilon > 0$ in an n vertex graph. Moreover, any algorithm that solves the above approximation problem in $n^{o(\log n)}$ time would yield an algorithm with essentially the same running time for the hidden clique problem, and there is no such known result despite the extensive study of the hidden clique problem (see [14, 5, 13, 12]).

Our strategy for approximating the densest k-by-k combinatorial rectangle follows the paradigm outlined in the introduction. First, note that the problem can be equivalently reformulated as follows.

▶ **Lemma 4.** *Let $A \in [0,1]^{m\times n}$. Then*

$$\max_{S,T:|S|=|T|=k} \text{density}(A_{S,T}) = \max \; x^T A y \quad x \in \Delta_m, y \in \Delta_n \quad \|x\|_\infty \leq 1/k, \|y\|_\infty \leq 1/k$$

Proof. For fixed y, the function $x^T Ay$ is linear in x and vice versa, thus it is easy to replace any solution by one of at least the same value in which each x_i and each y_j is either 0 or $1/k$. This corresponds to the problem of maximizing the quantity density($A_{S,T}$) over all sets S of k rows and T of k columns. ◀

By Lemma 4 it can be seen that the densest k-by-k combinatorial rectangle fits into the general class of problems of our approximation algorithm. Thus, as described above, by iterating over elements of the cover and sequentially solving the associated linear programs, we obtain the following theorem.

▶ **Theorem 5.** *Let $A \in [0,1]^{m\times n}$. Suppose that there is a deterministic (or Las Vegas randomized) algorithm to enumerate an $\epsilon/2$-net for A in time t. Then a solution to the k-by-k densest combinatorial rectangle within an additive ϵ of the optimal can be found in time $t \cdot \mathrm{poly}(mn)$.*

3 γ_2 Bounds on the Cover Number

Results of [7] show that an ϵ-net for A can be constructed by a randomized algorithm in time $(1/\epsilon)^{O(d)}$ given a matrix B of rank d that is an $\epsilon/2$-approximation of A. A drawback to this result is that it requires finding such a low rank approximation B.

We address this issue here by considering the γ_2 norm. As we describe next, the (approximate) γ_2 norm characterizes the approximate rank up to a logarithmic factor in the size of the matrix and small change in the error parameter [23]. Moreover, the approximate γ_2 norm can be computed in polynomial time via semidefinte programming, and thus also gives a polynomial time randomized Las Vegas algorithm to find an approximation B whose rank is within a logarithmic factor of the optimal. Combining this with the result of [7] gives a randomized Las Vegas algorithm for constructing an ϵ-net for an m-by-n matrix A of approximate rank d working in time $(1/\epsilon)^{O(d\log(mn))}$. We also give a simple and direct proof of a weaker result solely in terms of the γ_2 norm. Namely, if $\gamma = \min_{B:\|A-B\|_\infty \le \epsilon/4} \gamma_2(B)$ then there is a randomized algorithm constructing an ϵ-net for A in time $(\gamma/\epsilon)^{\gamma^2 \log(mn)/\epsilon^2}$.

3.1 Factorization Norm

For a m-by-n matrix A of rank d, let $\sigma_1(A) \ge \cdots \ge \sigma_d(A) \ge 0$ denote the non-zero singular values of A. The trace norm $\|A\|_{tr} = \sum_{i=1}^{d} \sigma_i(A)$ is the sum of the singular values of A. A simple bound on the rank of A can be given in terms of the trace norm,

$$\|A\|_{tr} = \sum_{i=1}^{d} \sigma_i(A) \le d^{1/2} \left(\sum_{i=1}^{d} \sigma_i^2(A) \right)^{1/2}.$$

This gives

$$\mathrm{rank}(A) \ge \left(\frac{\|A\|_{tr}}{\|A\|_F} \right)^2, \tag{2}$$

where $\|A\|_F = \sqrt{\sum_i \sigma_i(A)^2} = \sqrt{\mathrm{Tr}(AA^*)}$ is the Frobenius norm of A.

A drawback to this bound is that it is non-monotone in the sense that it can give a better bound on a submatrix of A than on A itself. We can remedy this in the following way. Let $A \circ B$ denote the entrywise product of A and B. As $\mathrm{rank}(A) \ge \mathrm{rank}(A \circ uv^*)$ for any vectors u, v we can maximize the above bound on $A \circ uv^*$ over all vectors u, v. This motivates the definition of γ_2. Here $\|u\|$ denotes the ℓ_2 norm of u.

▶ **Definition 6.**

$$\gamma_2(A) = \max_{\substack{u,v \\ \|u\|=\|v\|=1}} \|A \circ uv^*\|_{tr}$$

In a similar way to rank, we can define an approximate version of γ_2. Originally this was defined in a multiplicative sense [25], but for consistency with approximate rank we define it in an additive way here.

▶ **Definition 7.** Let A be a matrix and $\epsilon \geq 0$.

$$\gamma_2^\epsilon(A) = \min_{\substack{B \\ \|A-B\|_\infty \leq \epsilon}} \gamma_2(B).$$

Exactly as in (2) we obtain that γ_2^ϵ gives the following lower bound on approximate rank.

▶ **Theorem 8.** *Let A be a matrix and $\epsilon \geq 0$. Then*

$$\mathrm{rank}_\epsilon(A) \geq \left(\frac{\gamma_2^\epsilon(A)}{\|A\|_\infty + \epsilon}\right)^2.$$

To show that γ_2^ϵ is also not too much smaller than the approximate rank it is useful to work with an alternative characterization of γ_2 as a factorization norm. Let $\|v\|_p$ denote the ℓ_p norm of v. For a m-by-n matrix A and non-negative integers p, q (possibly ∞) define the norm

$$\|A\|_{p\to q} = \max_{\substack{y \\ \|y\|_p = 1}} \|Ay\|_q .$$

By writing γ_2 as a semidefinite program and taking the dual, one arrives at the following formulation (see, for example, [10] or [26]).

▶ **Lemma 9.** *Let A be an m-by-n matrix. Then*

$$\gamma_2(A) = \min_{\substack{X,Y \\ XY=A}} \|X\|_{2\to\infty}\|Y\|_{1\to 2}.$$

Notice that $\|X\|_{2\to\infty}$ is equal to the largest ℓ_2 norm of a row of X. Similarly $\|Y\|_{1\to 2}$ is equal to the largest ℓ_2 norm of a column of Y.

Using the Johnson-Lindenstrauss [15] dimension reduction lemma, [23] show that the approximate γ_2 norm in fact characterizes the approximate rank, up to a logarithmic factor and small change in the approximation parameter.

▶ **Theorem 10** ([23])**.** *Let A be an m-by-n matrix with $\gamma_2^\epsilon(A) = \gamma$ witnessed by a factorization $A = XY$ where X is an m-by-k matrix and Y is k-by-n. For any $\delta > 0$ let $r = 8\gamma^2 \ln(4mn)/\delta^2$. Then*

$$\Pr_R[\|A - XRR^TY\|_\infty \leq \delta] \geq \frac{1}{2},$$

where the probability is taken over R a random k-by-r matrix with entries independent and identically distributed according to the normal distribution with mean 0 and variance 1. In particular,

$$\mathrm{rank}_{\delta+\epsilon}(A) \leq 8\ln(4mn)\frac{\gamma_2^\epsilon(A)^2}{\delta^2}$$

The logarithmic factor in this theorem is in fact necessary, as can be seen with the identity matrix. The identity matrix I_n of size n has $\gamma_2(I_n) = 1$, but Alon [6] shows that $\mathrm{rank}_\epsilon(I_n) = \Omega(\frac{\log(n)}{\epsilon^2 \log(1/\epsilon)})$ for $\frac{1}{2\sqrt{n}} \leq \epsilon \leq \frac{1}{4}$.

Theorem 10 combined with the results in [7] gives the following corollary.

▶ Corollary 11. Let A be an $m \times n$ matrix with entries in $[-1, 1]$ and $\mathrm{rank}_{\epsilon/4}(A) = d$. Then an ϵ-net for A can be constructed by a Las Vegas randomized algorithm in time $(1/\epsilon)^{O(d\ln(mn))}$.

3.2 Constructing ϵ-Nets via γ_2

In this section we prove from scratch an upper bound on the covering number in terms of the γ_2 norm. This gives weaker bounds than Corollary 11 but has the advantage of having a direct and simple proof.

▶ **Theorem 12.** *Let A be an m-by-n matrix. Suppose that $A = XY$ where X is m-by-d, Y is d-by-n, and $\|X\|_{2\to\infty}\|Y\|_{1\to 2} = \gamma$. Then $N_\epsilon(A) = O(\gamma/\epsilon)^d$. Moreover, an ϵ-net of this size can be constructed in time $O(\gamma/\epsilon)^d \mathrm{poly}(mn)$.*

Proof. We can assume without loss of generality that $\|X\|_{2\to\infty} = \gamma$ and $\|Y\|_{1\to 2} = 1$. Then by definition $\|Yx\|_2 \le 1$ for any x with $\|x\|_1 \le 1$.

Let S be an ϵ/γ-net for the unit ball in $\mathbb{R}^d$ of size $O(\gamma/\epsilon)^d$. There are standard explicit constructions for such nets that work in time $O(\gamma/\epsilon)^d$, for example by taking a tiling by cubes of size $\epsilon/(\gamma\sqrt{d})$. Then

$$\forall x : \|x\|_1 = 1, \exists\ \tilde{y} \in S : \|Yx - \tilde{y}\|_2 \le \frac{\epsilon}{\gamma}\,.$$

Now, apply X to the vector $Yx - \tilde{y}$. Since $\|X\|_{2\to\infty} = \gamma$, it holds that

$$\|XYx - X\tilde{y}\|_\infty \le \gamma \cdot \frac{\epsilon}{\gamma} = \epsilon\,.$$

Thus we can take the set $T = \{X\tilde{y} : \tilde{y} \in S\}$. This can be constructed from S in time $O(\gamma/\epsilon)^d \mathrm{poly}(m,n)$. ◀

▶ Corollary 13. Let A be an m-by-n matrix and $\epsilon > 0$. Let $\gamma_2^{\epsilon/4}(A) = \gamma$. Then $N_\epsilon(A) = (\gamma/\epsilon)^{O(\gamma^2 \ln(mn)/\epsilon^2)}$. Moreover, an ϵ-net of this size can be constructed by a Las Vegas randomized algorithm in time $(\gamma/\epsilon)^{O(\gamma^2 \ln(mn)/\epsilon^2)}\mathrm{poly}(m,n)$.

Proof. First we solve the semidefinite program for $\gamma_2^{\epsilon/4}$ to obtain matrices U, V such that $\|UV - A\|_\infty \le \epsilon/4$ and $\|U\|_{2\to\infty}\|V\|_{1\to 2} = \gamma$. Then let $X = UR$ and $Y = R^T V$ for a random d-by-d matrix R with $d = O(\frac{\gamma^2}{\epsilon^2}\ln(mn))$. By Theorem 10 with high probability we have $\|A - XY\|_\infty \le \epsilon/2$. Applying Theorem 12 to XY gives a set T of size $O(\frac{\gamma}{\epsilon})^d$ such that

$$\forall x \in \Delta_n, \exists \tilde{x} \in T : \|XYx - \tilde{x}\|_\infty \le \frac{\epsilon}{2}\,.$$

Thus

$$\begin{aligned}\|Ax - \tilde{x}\|_\infty &= \|Ax - XYx + XYx - \tilde{x}\|_\infty \\ &\le \|(A - XY)x\|_\infty + \|XYx - \tilde{x}\|_\infty \\ &\le \epsilon\ .\end{aligned}$$

◀

The identity matrix again shows that the logarithmic factor in the statement of Corollary 13 is necessary.

▶ **Lemma 14.** *Fix a natural number $k > 0$. Then $N_\epsilon(I_n) \ge \binom{n}{k}$, for every $\epsilon < \frac{1}{2k}$.*

Proof. For a subset $S \subseteq [n]$ of size k denote by v_S the vector $v = (v_1, v_2, \ldots, v_n)$ satisfying $v_i = 1/k$ if $i \in S$ and $v_i = 0$ otherwise. Then, for every pair of subsets $S \ne T \subseteq [n]$ of size k, we have that $\|v_S - v_T\|_\infty = 1/k$. If $\epsilon < \frac{1}{2k}$ this implies that v_S and v_T must have distinct representatives, which implies the lemma. ◀

4 A Quasi-polynomial Upper Bound and VC dimension

Considering the upper bounds on the cover number in terms of approximate rank or approximate γ_2 described above, one might build the expectation that these bounds characterize the cover number well. This is actually true for some ranges of error, as we will see in Section 5.1. But for fixed ϵ this is far from the truth. In this case, by the results in [3] the bound via approximate rank is at least as large as $2^{\Omega(n)}$ for almost all $n \times n$ sign matrices, and the same holds for the bound via approximate γ_2, while on the other hand, the next theorem states that the cover number is at most $n^{O(\log n)}$ for every such matrix.

▶ **Theorem 15.** *For any* $A \in [-1,1]^{m\times n}$,

$$N_\epsilon(A) \leq \binom{n + \frac{2\ln(2m)}{\epsilon^2}}{\frac{2\ln(2m)}{\epsilon^2}} < n^{\frac{2\ln(2m)}{\epsilon^2}}.$$

Theorem 15 can be derived as a special case of Maurey's Lemma ([27]), see also [28, Lemma 13] for a related result. For completeness we include here a short proof.

Proof. Put $k = \frac{2\ln(2m)}{\epsilon^2}$, and let A be as in the theorem. Let $A_1, A_2, \ldots, A_n$ denote the columns of A. It suffices to prove that for any vector $y = (y_1, y_2, \ldots, y_m)$ in the convex hull of the columns of A there is a (multi)-subset $S = (A_{i_1}, A_{i_2}, \ldots, A_{i_k})$ of k (not necessarily distinct) columns of A so that

$$\|\frac{1}{k}\sum_{j=1}^{k} A_{i_j} - y\|_\infty \leq \epsilon, \tag{3}$$

since the number of these subsets is at most

$$\binom{n + \frac{2\ln(2m)}{\epsilon^2}}{\frac{2\ln(2m)}{\epsilon^2}}.$$

To prove this fact suppose $y = \sum_{j=1}^{m} A_j p_j$. Choose the elements of the subset S randomly and independently among the columns of A (with repetitions), where each A_{i_j} is obtained by picking one of the columns, where A_j is chosen with probability p_j. The coordinate number i of the random sum $\sum_{j=1}^{k} A_{i_j}$ obtained is thus a sum of k independent identically distributed random variables, each having expectation y_i/k and each being bounded in absolute value by $1/k$. It thus follows by the standard Chernoff-Hoeffding-Azuma Inequality (c.f., e.g., [8]) that the probability this coordinate differs from y_i by more than ϵ is smaller than $1/m$, and hence with positive probability (3) holds. ◀

As we show in Section 5, the assertion of Theorem 15 is essentially tight. But it can still be improved if we have some extra information about the matrix A. One way to do it is in terms of the VC-dimension of A, defined next.

▶ **Definition 16.** Let $A \in \mathbb{R}^{m\times n}$ be a matrix. Let $C = \{c_1, \ldots, c_k\} \subseteq [n]$ be a subset of columns of A. We say that A shatters C if there are real numbers $(t_{c_1}, \ldots, t_{c_k})$ such that for any $D \subseteq C$ there is a row i with $A(i,c) < t_c$ for all $c \in D$ and $A(i,c) > t_c$ for all $c \in C \setminus D$. Let $\mathrm{VC}(A)$ be the maximal size of a set of columns shattered by A.

Note that $\mathrm{VC}(A) \leq \log(m)$ for any m-by-n matrix. Sometimes the quantity in Definition 16 is referred to as pseudo-dimension, and VC dimension is reserved for sign or boolean matrices

where the choice of thresholds is not needed. For convenience we will use VC dimension for this more general definition as well.

The key to our upper bound is the following lemma. This was originally shown, with an additional logarithmic factor, in the original paper of Vapnik and Chervonenkis defining VC dimension [30]. The logarithmic factor was later removed by Talagrand [29] (see also [18] for a simpler proof).

▶ **Lemma 17.** *[30, 29, 18] Let $A \in [-1,1]^{m\times n}$ be a matrix with $\mathrm{VC}(A) = d$. For $S \subseteq [n]$ let $\chi_S \in \{0,1\}^n$ denote its characteristic vector. For any $\epsilon > 0$ and $S \subseteq [n]$ there is a set $T \subset S$ of size $|T| = O\left(\frac{d}{\epsilon^2}\right)$ such that*

$$\left\| \frac{A\chi_S}{|S|} - \frac{A\chi_T}{|T|} \right\|_\infty \leq \epsilon\,.$$

This lemma says that every uniform combination of columns of A can be ϵ approximated by a uniform combination of about $\mathrm{VC}(A)/\epsilon^2$ many columns. In the next theorem we obtain an upper bound on the cover number in terms of VC dimension by reducing the case of arbitrary probability distributions to that of uniform distributions and applying Lemma 17.

▶ **Theorem 18.** *Let $A \in [-1,1]^{m\times n}$ be a matrix with $\mathrm{VC}(A) = d$. Then*

$$N_\epsilon(A) \leq n^{O(d/\epsilon^2)}.$$

Proof. We will use Lemma 17 to show that every element of the convex hull of the columns of A can be ϵ-approximated by the average of some $O(d/\epsilon^2)$ columns (with repetition) of A.

Let $N = 2n/\epsilon$. Let A' be the matrix where every column of A is repeated N times. As duplicating columns does not change the VC dimension we have $\mathrm{VC}(A') = d$. Let $p \in [0,1]^n$ be a probability vector. Define $p' \in [0,1]^{Nn}$ as

$$p'(jN+i) = \begin{cases} \frac{1}{N} \text{ for } j = 0,\ldots,n-1 \text{ and } i = 1,\ldots,\lfloor p(j)N \rfloor \\ 0 \text{ for } j = 0,\ldots,n-1 \text{ and } i = \lfloor p(j)N \rfloor + 1,\ldots,N. \end{cases}$$

Note that $\|Ap - A'p'\|_\infty \leq \frac{n}{N} \leq \epsilon/2$. As p' is a normalized characteristic vector, by Lemma 17 we have that there is a set D of size $O\left(\frac{d}{\epsilon^2}\right)$ such that

$$\left\| A'p' - \frac{A'\chi_D}{|D|} \right\|_\infty \leq \epsilon/2\,.$$

Thus to obtain an ϵ-net for A it suffices to take all uniform combinations of $O\left(\frac{d}{\epsilon^2}\right)$ columns, taken with repetition. This gives the theorem. ◀

5 Lower Bounds

In this section, we show some lower bounds on the covering number in terms of approximate rank, one-way communication complexity and VC dimension. Some of the lower bounds we prove match the corresponding upper bounds shown earlier.

5.1 Tight Lower Bounds in the General Case

In this section, we show that $N_{0.99}(A) = n^{\Omega(\log n)}$ for a random sign matrix A, and thus that our upper bounds in terms of VC dimension is tight in this case. We also show a lower bound of $N_{2/7}(A) = 2^{\Omega(\mathrm{VC}(A))}$ for any sign matrix A. Our lower bounds follow from bounds

on the closely related *packing number* of A. Let $C_\delta(A)$ be the maximal number of δ-size ℓ_∞ balls that can be packed into the convex hull of the columns of A. Then

$$C_{2\epsilon}(A) \leq N_\epsilon(A) \leq C_\epsilon(A)\,.$$

We obtain our bounds via the next simple lemma, together with the existence of appropriate nearly disjoint families of sets.

▶ **Lemma 19.** *Let A be an m-by-n sign matrix and $\mathcal{F}$ a family of subsets of $[n]$ such that*
1. *for every $F, F' \in \mathcal{F}$ the columns of A in $F \cup F'$ are shattered.*
2. *$|F \cap F'| \leq (1-\delta/2)|F|$ for all distinct $F, F' \in \mathcal{F}$.*

Then $N_{\delta/2}(A) \geq |\mathcal{F}|$.

Proof. Let A_j denote the jth column of A. For any $F \in \mathcal{F}$, the vector

$$v_F = \frac{1}{|F|}\sum_{j\in F} A_j$$

lies in the convex hull of the columns of A. Now consider $\|v_F - v_{F'}\|_\infty$ for distinct $F, F' \in \mathcal{F}$. As $F \cup F'$ is shattered, there is a row i such that $A(i,j) = 1$ for all $j \in F$ and $A(i,j) = -1$ for all $j \in F' \setminus F$. Thus

$$\|v_F - v_{F'}\|_\infty = 1 - \frac{1}{|F'|}\left(|F \cap F'| - |F' \setminus F|\right) \geq \delta\,.$$

◀

▶ Claim 20. There is a family $\mathcal{F}$ of subsets of $[d]$ such that
1. $|F| \geq \frac{7}{16}d$ for all $F \in \mathcal{F}$
2. $|F \cap F'| \leq \frac{5}{16}d$ for all distinct $F, F' \in \mathcal{F}$
3. $|\mathcal{F}| \geq 2^{.001d}$

▶ Claim 21. There is a family $\mathcal{F}$ of subsets of $[n]$ such that
1. $|F| = 0.49\log_2 n$ for all $F \in \mathcal{F}$
2. $|F \cap F'| \leq 0.0001\log_2 n$ for all distinct $F, F' \in \mathcal{F}$
3. $|\mathcal{F}| \geq n^{\Omega(\log n)}$

Proof. The existence of such $\mathcal{F}$ follows either by a simple probabilistic argument, or by using known bounds for constant weight codes, or by an explicit constructions using polynomials. ◀

▶ **Lemma 22.** *Let A be a sign matrix. Then $N_{2/7}(A) \geq 2^{\Omega(\mathrm{VC}(A))}$.*

Proof. This follows from Lemma 19 together with the set family from Claim 20. ◀

▶ **Lemma 23.** *For almost all n-by-n sign matrices A,*

$$N_{.99}(A) = n^{\Omega(\log n)}\,.$$

Proof. Let $A = (a_{ij})$ be a random n-by-n sign matrix, where each entry $a_{ij} \in \{-1,1\}$ is chosen randomly, independently and uniformly in $\{-1,1\}$. We show that with high probability A shatters every subset $J \subseteq [n]$ of columns with $|J| \leq 0.98\log n$. Indeed, for a fixed J and sign pattern $s \in \{-1,+1\}^{|J|}$, the probability that no row of A restricted to J is equal to s is

$$(1 - 2^{-|J|})^n < e^{-n^{0.02}}.$$

The result thus follows from the union bound.

Therefore, the VC dimension of a random n-by-n sign matrix is greater than $0.98\log n$ with high probability. The proof of the Lemma now follows from Lemma 19 using the set family from Claim 21. ◀

5.1.1 An Explicit Example

In addition to the lower bound $N_{.99}(A) = n^{\Omega(\log(n))}$ for a random sign matrix A, we can also show explicit examples where the cover number is this large. We show next that a simple Hadamard matrix requires covers of quasi-polynomial size. Consider the 2^t-by-2^t Hadamard matrix, $H = (h_{v,M})$, whose columns are indexed by monomials $M = \prod_{i \in I} x_i$ with $I \subset [t]$ and whose rows are indexed by vectors $v \in \{-1, 1\}^t$, where $h_{v,M} = M(v)$. In this matrix, for any choice of monomials $M_1, M_2, \ldots, M_k$ in which no product of a subset is identically 1, the polynomial

$$\frac{1 + M_1}{2} \frac{1 + M_2}{2} \cdots \frac{1 + M_k}{2}$$

is the average of 2^k monomials. Its value on a vector v is 1 if $M_j(v) = 1$ for all j, and is 0 otherwise. This gives, if we shift to an additive rather than multiplicative notation, for every subspace of dimension $t/2$ of Z_2^t, a vector in the convex hull of the columns of H which is 1 on the members of the subspace and 0 outside it. Therefore, this example is an $n = 2^t$ by $n = 2^t$ sign matrix H for which $N_\epsilon(A) \geq n^{(1+o(1)) \log n/4}$ for all $\epsilon < 1/2$.

5.2 Lower Bounds on the Cover Number in Terms of Approximation Rank

For a vector v and a linear subspace U we also define

$$d(v, U) = \min_{u \in U} \|v - u\|_\infty .$$

▶ **Lemma 24.** *Let A be a real matrix and fix $0 < \epsilon$. Let d be the ϵ-approximate rank of A. Then there are d columns of A, $a_{i_1}, a_{i_2}, \ldots, a_{i_d}$, such that*

$$d(a_{i_j}, span\{a_{i_1}, \ldots, a_{i_{j-1}}\}) \geq \epsilon ,$$

for every $1 \leq j \leq d$.

Proof. We construct the set of columns inductively. We choose the first column as any nonzero column (such a column must exist if $d > 0$). If we have constructed d columns already, we are done. Otherwise we have $a_{i_1}, a_{i_2}, \ldots, a_{i_t}$ for $t < d$. By definition of approximate rank, and since $d_\epsilon(A) = d > t$, there must be a column that is ϵ-far from $\text{span}(a_{i_1}, a_{i_2}, \ldots, a_{i_t})$. We add this column to the set. ◀

▶ **Theorem 25.** *Let A be a real matrix and fix $0 < \epsilon$. Let d be the ϵ-approximate rank of A. Then*

$$N_{\epsilon/d}(A) = \Omega\left(\frac{2^d}{\sqrt{2d}}\right) .$$

Proof. By Lemma 24 there are d columns of A, $a_{i_1}, a_{i_2}, \ldots, a_{i_d}$, such that

$$d(a_{i_j}, \text{span}\{a_{i_1}, \ldots, a_{i_{j-1}}\}) \geq \epsilon,$$

for every $1 \leq j \leq d$. Assume w.l.o.g that these are the first d columns of A, $a_1, a_2, \ldots, a_d$. For simplicity we assume that d is even; if not the argument below can be done with $d - 1$ which only changes the bound by a constant factor.

Consider the set of vectors $S = \{\frac{2}{d} \sum_{i \in T} a_i : T \subseteq [d], |T| = d/2\}$. We claim that for every two vectors $v, u \in S$ it holds that $\|v - u\|_\infty \geq \epsilon/d$: Let $u = \frac{2}{d} \sum_{i \in T_1} a_i$ and $v = \frac{2}{d} \sum_{i \in T_2} a_i$

for $T_1 \neq T_2$. Then

$$u - v = \frac{2}{d}\left(\sum_{i\in T_1} a_i - \sum_{i\in T_2} a_i\right) = \frac{2}{d}\left(\sum_{i\in T_1\setminus T_2} a_i - \sum_{i\in T_2\setminus T_1} a_i\right).$$

Let j be the largest index in $T_1 \triangle T_2$. Assume w.l.o.g that $j \in T_1$, we have

$$\|u - v\|_\infty = \frac{2}{d}\|a_j - \left(\sum_{i\in T_2\setminus T_1} a_i - \sum_{j\neq i\in T_1\setminus T_2} a_i\right)\|_\infty \geq \frac{2\epsilon}{d}.$$

The last inequality is because $\sum_{i\in T_2\setminus T_1} a_i - \sum_{j\neq i\in T_1\setminus T_2} a_i$ is contained in the linear subspace spanned by $a_1, a_2, \ldots, a_{j-1}$.

Since S is in the convex hull of the columns of A and $|S| \geq 2^d/\sqrt{2d}$, we get that

$$\frac{2^d}{\sqrt{2d}} \leq C_{2\epsilon/d}(A) \leq N_{\epsilon/d}(A).$$

◀

5.3 Lower Bounds via Communication Complexity

▶ **Lemma 26.** *Let A be a sign matrix, and denote by $cc(A)$ the one-way (from Bob to Alice) deterministic communication complexity of A. Then*

$$cc(A) \leq \log(N_\epsilon(A))$$

for every ϵ in $(0, 1)$.

Proof. Let $k = cc(A)$, then there are 2^k distinct columns in A. Since the ℓ_∞ distance between every two distinct sign vectors is at least 2, we have

$$2^k \leq N_\epsilon(A)$$

for every $\epsilon \in (0, 1)$. ◀

The log rank conjecture, formulated by Lovász and Saks [22] is a long-standing open problem in communication complexity. A simple argument shows that $\log \mathrm{rank}(A)$ is a lower bound on the deterministic communication complexity $D(A)$ of A. The log rank conjecture states that this bound is polynomially tight, $D(A) = \log(\mathrm{rank}(A))^{O(1)}$. The best upper bound on communication complexity in terms of rank was recently improved to show $D(A) = O(\sqrt{\mathrm{rank}(A)}\log(\mathrm{rank}(A)))$ [21].

Combining Lemma 26 with the relation between the covering number and approximate rank proved in [7], we get an upper bound on the one-way communication complexity in terms of the *approximate* rank.

▶ Corollary 27. Let A be a sign matrix, and denote by $cc(A)$ the one-way deterministic communication complexity of A. Then

$$cc(A) \leq 3\mathrm{rank}_{1/2}(A) + O(1).$$

6 Conclusion and Open Problems

Efficiently enumerable covers of the convex hull of a matrix lead to efficient approximation algorithms for a broad class of optimization problems including Nash equilibrium and densest k-by-k combinatorial rectangle. We have shown that $N_\epsilon(A) \leq n^{O(\mathrm{VC}(A)/\epsilon^2)}$ and moreover that such covers can be deterministically enumerated in about the same time. This result unifies many previous approximation algorithms for Nash equilibrium in the literature, including the quasi-polynomial time approximation algorithm of Lipton et al. [19] and the approximation algorithm of Kannan and Theobald for game matrices A, B such that $A + B$ has constant rank [17]. For the densest k-by-k combinatorial rectangle problem this gives for the first time a $n^{O(\log(n)/\epsilon^2)}$ time algorithm to obtain an additive ϵ-approximation.

The central open problem if Nash Equilibrium has a polynomial time approximation scheme remains open. One avenue to make progress on this question may be to find a common generalization of the cover based approximation algorithms given here and in [7], with the approximation algorithm for random games of Bárány et al. [11].

Acknowledgements. We thank Gideon Schechtman, Shai Shalev-Shwartz and Santosh Vempala for helpful discussions. Noga Alon is supported in part by an ERC Advanced grant, a USA-Israeli BSF grant, an ISF grant, the Israeli I-Core program, and by the Simonyi Fund. Troy Lee is supported in part by the Singapore National Research Foundation under NRF RF Award No. NRF-NRFF2013-13.

References

1 N. Alon, S. Arora, R. Manokaran, D. Moshkovitz, and O. Weinstien. On the inapproximability of the densest k-subgraph problem. unpublished manuscript, 2011.

2 N. Alon, R. Duke, H. Lefmann, V. Rödl, and R. Yuster. The algorithmic aspects of the regularity lemma. *J. Algorithms*, 16(1):80–109, 1994.

3 N. Alon, P. Frankl, and V. Rödl. Geometric realization of set systems and probabilistic communication complexity. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*, pages 277–280. IEEE, 1985.

4 S. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for dense instances of np-hard problems. *Journal of Computer and System Sciences*, 58:193–210, 1999.

5 Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. In *SODA*, pages 594–598, 1998.

6 N. Alon. Perturbed identity matrices have high rank: proof and applications. *Combinatorics, Probability, and Computing*, 18:3–15, 2009.

7 N. Alon, T. Lee, A. Shraibman, and S. Vempala. The approximate rank of a matrix and its algorithmic applications. In *Proceedings of the 45th ACM Symposium on the Theory of Computing*. ACM, 2013.

8 N. Alon and J. Spencer. *The Probabilistic Method*. John Wiley and Sons, third edition, 2008.

9 A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: an $n^{1/4}$ approximation for densest subgraph. In *Proceedings of the 42nd ACM Symposium on the Theory of Computing*, pages 201–210, 2010.

10 R. Bhatia. *Positive Definite Matrices*. Princeton University Press, 2007.

11 Imre Bárány, Santosh Vempala, and Adrian Vetta. Nash equilibria in random games. *Random Struct. Algorithms*, 31(4):391–405, 2007.

12 Yael Dekel, Ori Gurel-Gurevich, and Yuval Peres. Finding hidden cliques in linear time with high probability. *CoRR*, abs/1010.2997, 2010.

13 Uriel Feige and Dorit Ron. Finding hidden cliques in linear time. In *AOFA*, 2010.

14 Mark Jerrum. Large cliques elude the metropolis process. *Random Struct. Algorithms*, 3(4):347–360, 1992.

15 W. Johnson and J. Lindenstrauss. Extensions of Lipschitz mapping into Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.

16 S. Khot. Ruling out PTAS for graph min-bisection, densest subgraph and bipartite clique. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*. IEEE, 2004.

17 R. Kannan and T. Theobald. Games of fixed rank: A hierarchy of bimatrix games. *Econom. Theory*, 42:157–173, 2010. appeared in SODA 2007.

18 Y. Li, P. M. Long, and A. Srinivasan. Improved bounds on the sample complexity of learning. *Journal of Computer and System Sciences*, 62:516–527, 2001.

19 R. J. Lipton, E. Markakis, and A. Mehta. Playing large games using simple strategies. In *ACM Conference on Electronic Commerce*, pages 36–41, 2003.

20 N. Linial, S. Mendelson, G. Schechtman, and A. Shraibman. Complexity measures of sign matrices. *Combinatorica*, 27(4):439–463, 2007.

21 S. Lovett. Communication is bounded by root of rank. Technical Report arXiv:1306.1877, arXiv, 2013.

22 L. Lovász and M. Saks. Möbius functions and communication complexity. In *Proceedings of the 29th IEEE Symposium on Foundations of Computer Science*, pages 81–90. IEEE, 1988.

23 T. Lee and A. Shraibman. An approximation algorithm for approximation rank. In *Proceedings of the 24th IEEE Conference on Computational Complexity*, pages 351–357. IEEE, 2008. arXiv:0809.2093 [cs.CC].

24 N. Linial and A. Shraibman. Learning complexity versus communication complexity. *Combinatorics, Probability, and Computing*, 18:227–245, 2009.

25 N. Linial and A. Shraibman. Lower bounds in communication complexity based on factorization norms. *Random Structures and Algorithms*, 34:368–394, 2009.

26 T. Lee, A. Shraibman, and R. Špalek. A direct product theorem for discrepancy. In *Proceedings of the 23rd IEEE Conference on Computational Complexity*, pages 71–80. IEEE, 2008.

27 B. Maurey. Théorémes de factorisation pour les opérateurs linéaires à valeurs dans les espaces l_p. *Astérisque*, (11):1–163, 1974.

28 S. Sabato, S. Shalev-Shwartz, N. Srebro, D. Hsu and T. Zhang Learning Sparse Low-Threshold Linear Classifiers *CoRR*, abs/1212.3276, 2012.

29 M. Talagrand. Sharper bounds for gaussian and empirical processes. *Annals Probab.*, 22:28–76, 1994.

30 V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory Probab. Appl.*, 16:264–280, 1971.

Network Design with Coverage Costs*

Siddharth Barman[1], Shuchi Chawla[2], and Seeun Umboh[3]

1 **California Institute of Technology, U.S.**
barman@caltech.edu
2 **University of Wisconsin – Madison, U.S.**
shuchi@cs.wisc.edu
3 **University of Wisconsin – Madison, U.S.**
seeun@cs.wisc.edu

Abstract

We study network design with a cost structure motivated by redundancy in data traffic. We are given a graph, g groups of terminals, and a universe of data packets. Each group of terminals desires a subset of the packets from its respective source. The cost of routing traffic on any edge in the network is proportional to the total size of the distinct packets that the edge carries. Our goal is to find a minimum cost routing. We focus on two settings. In the first, the collection of packet sets desired by source-sink pairs is laminar. For this setting, we present a primal-dual based 2-approximation, improving upon a logarithmic approximation due to Barman and Chawla (2012) [7]. In the second setting, packet sets can have non-trivial intersection. We focus on the case where each packet is desired by either a single terminal group or by all of the groups. This setting does not admit an $O(\log^{\frac{1}{4}-\gamma} g)$-approximation for any constant γ under a standard assumption; we present an $O(\log g)$-approximation when the graph is unweighted.

Our approximation for the second setting is based on a novel spanner-type construction in unweighted graphs that, given a collection of g vertex subsets, finds a subgraph of cost only a constant factor more than the minimum spanning tree of the graph, such that every subset in the collection has a Steiner tree in the subgraph of cost at most $O(\log g)$ that of its minimum Steiner tree in the original graph. We call such a subgraph a group spanner.

1998 ACM Subject Classification F.2.0 Analysis of Algorithms and Problem Complexity

Keywords and phrases Network Design, Spanner, Primal Dual Method, Traffic Redundancy

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.48

1 Introduction

Some of the classical applications of the theory of algorithms are in transportation and commodity networks: how should commodities be transported from where they are manufactured to where they are consumed? How should pipelines be laid to be most effective at balancing costs with requirements? These questions have spawned many basic problems and theorems in the area of approximation algorithms: network flow, traveling salesman, Steiner tree, flow-cut gaps, etc. Over time, solutions to these problems have come to be applied to a different class of networks, namely communication networks. At a basic level, the problems in communication networks are similar: how should data be routed from its sources to its destinations? How should networks be designed to be able to handle different kinds of workload and traffic patterns? However, the underlying commodity in these

* This work was partially supported by NSF grants CNS-0846025 and CCF-1101470. Siddharth Barman gratefully acknowledges the support of the Linde/SISL postdoctoral fellowship.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 48–63

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

networks—data—is fundamentally different from physical commodities. Unlike the latter, data can be compressed, encoded, or replicated, at virtually no cost. Network algorithms that do not exploit these properties fail to utilize the entire capacity of the network.

The last few years have seen a rapid growth in "content aware" network optimization solutions, both within the academic literature (see, e.g., [1, 26], and references therein) as well as in the form of commercial technologies [9, 24]. One of the functionalities that these technologies provide is to remove duplicate traffic from the network. Every router in the network equipped with such a technology keeps track of recently seen traffic. When duplicates are detected, a single copy of the duplicated data is sent forward along with a short message containing instructions for replication at the next router. This defines a cost function on every link in the network, where the cost of carrying data is proportional to the number (or total size) of *distinct* packets that the link carries; in other words, it is a *coverage function* over the set of traffic streams that use the link. We study network design problems within this context.

We consider the following framework. We are given a weighted network, and multiple *commodities*, each with a source and several possible destinations that we collectively call terminals. Each commodity is composed of a number of different data packets drawn from a universe of packets; we call these sets of packets *demands*. Importantly, there is redundancy in traffic—different commodities may overlap in the sets of packets they contain, and so can benefit from using common routes. Our goal is to find a minimum cost routing for the given traffic matrix, assuming that we can buy bandwidth at a fixed rate on every edge. Formally, our solution specifies for each commodity a routing tree spanning all of the terminals for this commodity. The cost of this solution on any particular edge is proportional to the total size of the distinct packets that the edge carries. This problem was introduced in [7] where it was called redundancy aware network design.

Network design with coverage costs displays the same short-routes-versus-shared-routes tradeoff present in several classical network design problems with nonlinear costs, such as rent-or-buy network design [19, 15], access network design [4], and buy-at-bulk network design [5, 16, 21, 28]. However there are fundamental differences. The buy-at-bulk cost model is inspired by economies of scale in a physical commodity network—the volume of traffic that an edge carries is the sum of the volumes that the different commodities impose on it and the routing cost on the edge is a concave function of the total volume of traffic. On the other hand, in our setting, the volume of traffic itself is lowered due to the inherent nature of data traffic. In particular, this means that the savings achieved depend on the contents of the traffic and not just its quantity. We not only need to bundle traffic streams as much as we can, but we also need to decide the right sets of traffic streams to bundle. Consequently, the approximability of the problem also depends on the extent and manner in which different commodities share packets. When every source-sink pair in the network demands a distinct packet, that is, there is no data redundancy in the network, the problem reduces to finding the shortest route for each pair. When all of the demands are identical, the problem reduces to finding a single optimal Steiner forest over all of the terminal sets.

Laminar and sunflower demands. In this paper we focus on two special cases of the network design problem with coverage costs—the *laminar demands* setting, and the *sunflower demands* setting. In the laminar demands setting the packet sets corresponding to the commodities form a laminar family: the packet sets of any two commodities are either completely disjoint or one contains the other. There is a natural hierarchy over commodities in this setting and any commodity can use for free an edge that is being used for another commodity

that "dominates" it. So we may favor long routes for a commodity if those routes share edges with a dominating commodity, in comparison to shorter ones that do not share edges. Less intuitively, it may be useful to pick similar routes for two commodities with disjoint packets sets if a portion of the shared route can be used for a commodity that dominates both. Consequently, commodities that are higher up in the hierarchy are in some sense more important than commodities that are lower in the hierarchy.

Non-laminar settings, where packet sets can have arbitrary intersection, also display sharing of paths among similar as well as dissimilar commodities. However, we cannot exploit any natural ordering over commodities in determining which paths to use so we require techniques that are very different from those used for the laminar demands setting. As a first step, we study the simplest setting that captures the complexity introduced by non-trivial intersections. In the *sunflower demands* setting, every collection of demands has the same intersection. In other words, there is a common set of packets that belongs to every commodity, and every other packet belongs to exactly one commodity. A simple example of this setting is where each demand is of the form $\{0, i\}$; here 0 denotes the common packet, and i denotes the packet belonging only to commodity i. Once again our goal is to construct a routing tree for each commodity of minimum total cost. The cost of the collection of routing trees has two components. The first corresponds to the total size of the union of the routing trees: we pay for the cost of routing the common packets on this entire subgraph. The second corresponds to the costs of the individual trees, weighted by the sizes of their respective unique packets. A natural interpretation for the cost structure is as follows: for each edge, there is a fixed cost (per unit length) for buying the edge before it can be used for routing, and a variable cost (per unit length) that depends on the number of commodities being routed on it.

1.1 Our results and techniques

We present a primal-dual based 2-approximation for the laminar demands setting, improving upon a logarithmic approximation by [7] and matching the approximation factor for the Steiner forest problem, which is a special case. The sunflower demands setting, on the other hand, is much harder. In particular, it captures as a special case the buy-at-bulk network design problem with a single cable type with linear cost; this special case was shown in [2] to be inapproximable to within a factor polylogarithmic in g under standard assumptions, where g is the number of commodities (see Theorem 8 in Section 4). We present an $O(\log g)$ approximation for this problem under two further assumptions[1]: (1) the graph is unweighted; (2) every node is a terminal. We leave open the question of designing an approximation for the general sunflower demands setting. Note, however, that an $O(\log n)$ approximation can be obtained by first embedding the network into a tree with low distortion and then solving the problem on the tree; Here n is the number of nodes in the network. Our $O(\log g)$ approximation is based on a novel spanner-type construction described below that may be of independent interest. We now describe our techniques in more detail.

[1] We note that the first assumption by itself, i.e. the graph is unweighted, is without loss of generality: since our approximation is with respect to the total cost of the solution, and not with respect to the number of edges in it, we can break up each long edge into edges of equal size by introducing new nodes. However, the additional assumption that every vertex belongs to some terminal set disallows this sort of transformation.

Sunflower demands. A standard approach in network optimization is to approximate a given network by a subgraph that is much cheaper or sparser than the entire graph, and yet faithfully captures some essential property of the graph. For example, spanners [23] are low-cost subgraphs that approximately capture shortest path distances between every pair of points in the graph. Likewise, cut- and flow-sparsifiers [22, 20] are sparse subgraphs that approximate cuts and flows in the graph respectively. Network design with coverage costs defines another such graph sparsification problem that may be of independent interest. In particular, for a given solution to the network design problem, consider partitioning the edges into sets that carry a particular packet. Each such set is a Steiner forest over the terminal sets that demand that packet. Our goal is to find a solution that minimizes a weighted sum of the sizes of these Steiner forests. One way of doing so may be to find a subgraph that induces Steiner forests over each respective set of terminals corresponding to a single packet, that are simultaneously approximately minimal for their corresponding instances. This approach is particularly relevant for the sunflower demands setting. In that setting, the Steiner forest corresponding to the common packets is the entire subgraph itself, whereas the forest corresponding to packets unique to a commodity is simply the routing tree constructed for that commodity. We therefore ask: is there a subgraph that $\alpha-$approximates the size of the minimum Steiner forest over the union of all terminal sets, and at the same time induces a Steiner tree over each individual terminal set that is within a factor of β of the smallest such tree? We call such a subgraph an (α, β) *group spanner*. Group spanners generalize spanners: if for every pair of nodes in the graph our instance contains a terminal set comprising of the two nodes, then a group spanner for the instance simultaneously approximates the shortest path distances between every pair of nodes. The factor β is called the stretch of the spanner.

The main technical component in our approach for the sunflower demands setting is a construction for group spanners in unweighted graphs where the union of all terminal sets spans the entire graph. Our construction achieves an $(O(1), O(\log g))$ approximation. This implies an $O(\log g)$ approximation for the sunflower demands setting under those assumptions. A widely-believed conjecture of Erdős [11] and others (see [29] for a longer discussion) implies that no (α, β) group spanners with $\alpha\beta = o(\log g)$ exist, so our construction achieves the optimal tradeoff between size and stretch. The problem of extending our construction to arbitrary weighted graphs remains an interesting open question.

Laminar demands. To form intuition for this setting consider an instance with k different packets and $k+1$ commodities: for $i \leq k$ the demand set of commodity i contains only packet i, and demand set of commodity $k+1$ contains all of the k packets. Suppose also that every commodity has a single source and a single sink. Then, one approach to solving the problem is to first find a least cost path for commodity $k+1$, and then find least cost paths for the remaining commodities using the edges in the first path for free. This approach misses solutions where a slightly longer path for commodity $k+1$ is much more cost efficient for the remaining commodities than the shortest path for $k+1$. An alternative is to first find shortest paths for commodities 1 through k, and then find the least cost path for commodity $k+1$ that can use edges in previously picked paths at a cheaper cost. This misses solutions where picking slightly longer paths for commodities 1 through k leads to a greater sharing of the edges. The first approach is indeed the approach analyzed in [7] for the special case of the problem where there is a single source that belongs to all of the terminal sets. That paper shows that in any single source laminar demands setting routing commodities in order of decreasing sizes of demand sets achieves an $O(\log k)$ approximation where k is the number

of different packets in the universe[2].

We extend and improve the result of [7] to obtain a 2-approximation for the laminar demands setting with arbitrary terminal sets. Our approach is a hybrid of the two described above and is based on a non-standard LP formulation of the problem. Our LP encodes for each demand set the edges that carry this demand set but no other demand set that dominates it. We then apply a primal-dual approach. At a high level, we first consider commodities in *increasing* order of the sizes of their demand sets. However, instead of committing to a single path for each commodity before considering the next, we keep around a collection of all possible near-optimal paths for the smaller demand sets before considering choices for the larger demand sets. Then in a second pass, we finalize a single path (tree) for each commodity, considering commodities in *decreasing* order of sizes of their demand sets. That is, we commit to paths for the larger demand sets before finalizing paths for the smaller demand sets. The duals constructed for each commodity give a succinct description of all possible short paths connecting the source and the sink for that commodity. After having constructed all of the duals, we perform a reverse delete step that finalizes paths for commodities starting from the one with the largest demand and moving on to smaller demand sets.

1.2 Connections to other network optimization problems

The cost structure in the network design problem we consider is uniform in the sense that costs on different edges are related through constant factors. Obtaining a randomized $O(\log n)$ approximation for network design problems with a uniform cost structure is often easy: we can use the tree embeddings of Bartal [8] and Fakcharoenphol et al. [12] to convert the graph into a distribution over trees such that distances between nodes are preserved to within logarithmic factors in expectation. Then the expected cost of the optimal routing over the (random) tree is related within logarithmic factors to the cost of the optimal routing over the graph. Moreover, the problem is easy to solve on trees, because there is a unique path between every pair of nodes. We achieve much better approximation factors. For the laminar demands setting, we obtain a 2-approximation. For the sunflower demands setting, our approximation factor is $O(\log g)$; note that g is always at most n, and in most applications should be much smaller.

As mentioned earlier, network design with coverage costs is closely related but incomparable to other models of network design with uniform costs that display economies of scale. This includes, e.g., the uniform buy-at-bulk [5, 16, 21, 28], rent-or-buy [19, 15], and access network design [4, 14] problems. For all of these problems constant factor approximations are known in the uniform costs setting for the special case where all of the commodities share a common source. In the multi-commodity setting, i.e., with distinct sources and sinks, the rent-or-buy network design problem admits a 2-approximation [19, 15], but the buy-at-bulk network design problem is hard to approximate within poly-logarithmic factors [3].

Cost models specific to communication networks have been considered before in network design. Hayrapetyan et al. [17] study a single-source network design problem in which the cost on an edge is a monotone submodular function of the commodities that use the edge. They obtain an $O(\log n)$ approximation via tree embeddings [8, 12], where n is the number of vertices in the graph. The cost structure that we consider is a special case of the one

[2] In fact, after a slight transformation of the instance, the same approximation can be obtained by approximating the minimum Steiner tree for every demand set separately and combining the solutions

in [17] (coverage functions are submodular). However, unlike [17] we assume that terminals sets are arbitrary (in particular, they do not share a common source). Moreover, we obtain stronger approximation guarantees.

Shmoys et al. [25] study a facility location problem with a cost structure very similar to that in our sunflower demands setting. In their model, the cost of opening a facility has two components: a fixed cost (similar to the cost of routing the common packets in our setting), and a service specific cost (similar to the cost of routing other packets in our setting). They present a constant factor approximation for facility location with this cost structure. Svitkina and Tardos [27] further extend this to a facility location problem with hierarchical costs, again presenting a constant factor approximation. Extending our results to more general non-laminar coverage functions including hierarchical costs is an interesting open problem.

As mentioned earlier, a main component in our approach for the sunflower demands setting is a construction for group spanners in unweighted graphs. Group spanners generalize graph spanners. Low-stretch spanners have a number of applications, including distributed routing using small routing tables and in computing near-shortest paths in distributed networks (see [23] and references therein). In unweighted graphs it is well known that the size of the smallest spanner with multiplicative stretch k is equal to the maximum number of edges in a graph with girth at least $k+1$; this is known to be $O(n^{1+O(1/k)})$, and is conjectured tight. Our result is consistent with this bound: when the number of commodities g is equal to the number of vertex pairs, we get an $O(\log g) = O(\log n)$ stretch with a spanner of size $O(n)$. Other work on spanners has focused on additive stretch and weighted graphs (see, e.g., [10, 23, 30]).

Group spanners also generalize shallow-light spanning trees. The latter is a subgraph that is simultaneously an approximately-minimum spanning tree of the given graph, as well as an approximate-shortest-paths tree with respect to a given source node. Consider an instance with a special source node s that for every node v in the graph contains the terminal set $\{s, v\}$. Then an (α, β) group spanner for this instance simultaneously approximates the shortest path distance from s to v for every v to within a factor of β, and has size no more than α times the size of the minimum spanning tree in the graph. However, while our approach only guarantees $\beta = O(\log n)$ for $g = n$ commodities, it is possible to obtain an $(O(1/\epsilon), 1+\epsilon)$ approximation for any $\epsilon > 0$ [6, 18].

2 Problem Definition

In this section, we formally define Network Design with Coverage Costs. We are given a graph $G = (V, E)$ with costs c_e on edges, a universe Π of packets, and g commodities with terminal sets $X_1, \ldots, X_g \subseteq V$. The demand set of terminal set X_j is denoted $D_j \subseteq \Pi$, and we denote the collection of all demand sets as $\mathcal{D}$. A solution consists of a collection of g Steiner trees $\mathcal{T} = \{T_1, \ldots, T_g\}$ where T_j is a Steiner tree spanning terminal set X_j. The trees specify how packets are to be routed over the edges: the packets of demand D_j are routed over edges of T_j. For a solution $\mathcal{T}$, the load on edge e is $\ell_e(\mathcal{T}) = |\bigcup_{i:e\in T_i} D_i|$, i.e. the total number of distinct packets being routed over edge e. More generally, we can consider a setting in which packets have weights and we define the load on an edge to be the total weight of all of the distinct packets that an edge carries. The performance and running times of both of our algorithms are independent of the number of distinct packets, so we may assume without loss of generality that all packets have unit weight. Our goal is to find a solution $\mathcal{T}$ so as to minimize the total cost $\sum_{e\in E} c_e \ell_e(\mathcal{T})$.

We now describe the two special cases of network design with coverage costs that we

study. In the following, for a subgraph H, we write $c(H)$ for the total cost of edges in H, i.e. $c(H) := \sum_{e \in H} c_e$.

Laminar demands. In this setting, the collection of demand sets is laminar: for any $D, D' \in \mathcal{D}$, $D \cap D' \neq \emptyset$ implies either $D \subseteq D'$ or $D' \subseteq D$. In this case we can transform our objective into a simpler form where the cost of each edge is charged to a collection of *disjoint* demand sets. In particular, given a solution $\mathcal{T}$, for an edge e consider the demand sets D that are maximal among the collection $\{D_j : e \in T_j\}$ of demand sets that this edge carries. Because of laminarity, these maximal demand sets are disjoint, and so the load on the edge is simply the sum of the sizes of these demand sets. Accordingly, let us define $H_D(\mathcal{T})$ to be the set of edges e such that D is a maximal set in $\{D_j : e \in T_j\}$. The packet set D will contribute to the load on these edges. Then we can write the total cost of the solution $\mathcal{T}$ as

$$\ell(\mathcal{T}) = \sum_e c_e \ell_e(\mathcal{T}) = \sum_e \sum_{D : H_D(\mathcal{T}) \ni e} c_e |D| = \sum_D |D| \sum_{e \in H_D(\mathcal{T})} c_e = \sum_D |D| c(H_D(\mathcal{T})).$$

Further note that in a feasible solution $\mathcal{T}$, for each commodity j, the subgraph $\bigcup_{D \supseteq D_j} H_D(\mathcal{T})$ contains the tree T_j and therefore spans the terminal set X_j. Therefore, instead of specifying a Steiner tree for each terminal set, it suffices to specify a forest H_D for each demand set D such that each terminal set X_j is connected in $\bigcup_{D \supseteq D_j} H_D$.

Sunflower demands. In this setting, there is a special set of packets $P \subseteq \Pi$ such that for all $i \neq j$, we have $D_i \cap D_j = P$. In other words, $D_j = P \cup P_j$ with $P_i \cap P_j = \emptyset$ for all $i \neq j$. We can again transform our objective into a simpler form. For a routing solution $\mathcal{T} = \{T_1, T_2, \ldots, T_g\}$, let H denote the subgraph obtained by taking the union of the T_js. Observe that H is a Steiner forest for $X_1, \ldots, X_g$. We have to route P over H, since all terminal sets demand P, and P_j over T_j. Thus the cost of the routing solution can be expressed as $\ell(\mathcal{T}) = |P| c(H) + \sum_j |P_j| c(T_j)$.

We will now describe a lower bound on the cost of the optimal solution in this setting. For a vertex set X and subgraph H, let $\mathrm{St}_H(X)$ denote the cost of an optimal (i.e., minimum cost) Steiner tree over X in H. Let $\mathcal{T}^* = \{T_1^*, T_2^*, \ldots, T_g^*\}$ be an optimal routing solution to the given instance and let $H^* = \bigcup_j T_j^*$. Suppose F^* is an optimal Steiner forest for $X_1, \ldots, X_g$. Since H^* is a Steiner forest for $X_1, \ldots, X_g$ and T_j^* is a Steiner tree for X_j, we have $c(H^*) \geq c(F^*)$ and $c(T_j^*) \geq \mathrm{St}_G(X_j)$. Therefore the optimal routing-solution cost can be bounded as $\ell(\mathcal{T}^*) \geq |P|\ c(F^*) + \sum_j |P_j|\ \mathrm{St}_G(X_j)$.

Group spanners. For a graph $G = (V, E)$ with cost c_e on edges and g terminal sets $X_1, \ldots, X_g \subseteq V$, we say that subgraph H is an (α, β) *group spanner* if $c(H) \leq \alpha c(F^*)$ and $\mathrm{St}_H(X_j) \leq \beta\, \mathrm{St}_G(X_j)$ for all j. Here F^* denotes an optimal Steiner forest for $X_1, \ldots, X_g$ in G. Note that a group spanner generalizes the notion of a spanner since the latter asks for a sparse spanning subgraph H such that for every pair of vertices (u, v) we have β stretch: $d_H(u, v) \leq \beta d_G(u, v)$. Here $d_H(u, v)$ (respectively, $d_G(u, v)$) denotes the distance, with edge lengths c_e, between vertices u and v in H (respectively, G).

The following lemma shows that a good group spanner implies an approximation for the sunflower demands setting.

▶ **Lemma 1.** *Given an (α, β) group spanner H for graph G and terminal sets $X_1, X_2, \ldots, X_g$, we can obtain an $\alpha + 2\beta$ approximation for any sunflower demands instance defined over G and X_js.*

Proof. For all j, let H_j be the Steiner trees over X_j in H obtained via the MST heuristic [31]. We set $\{H_1, H_2, \ldots, H_g\}$ as the routing solution for the sunflower demand instance. The cost of this solution is no more than $|P|c(H) + \sum_j |P_j|c(H_j)$. Recall that the optimal routing-solution cost for sunflower demand instance is at least $|P|\ c(F^*) + \sum_j |P_j|\ \mathrm{St}_G(X_j)$. Therefore, using the fact that H is an (α, β) group spanner and $c(H_j) \leq 2\,\mathrm{St}_H(X_j)$ (guarantee of the MST heuristic) we get the desired claim. ◀

Note that using group spanners we get an oblivious approximation in the sense that the construction uses only the knowledge of the underlying graph and the terminal sets but not the demand sets.

In Section 4 we consider unweighted graphs with terminal sets that satisfy $V = \bigcup_j X_j$. We develop an algorithm that obtains a $(14, O(\log g))$ group spanner for such an instance, and so by Lemma 1 gives an $O(\log g)$ approximation to the sunflower demands setting over the instance (see Theorem 10).

3 A 2-approximation for the laminar demands setting

Recall that in the laminar demands setting, for all $D, D' \in \mathcal{D}$ with $D \cap D' \neq \emptyset$, we have $D \subseteq D'$ or $D' \subseteq D$. As established in Section 2, in order to obtain a feasible solution in this setting, it suffices to specify a forest H_D for each demand set D such that each terminal set X_j is connected in $\bigcup_{D \supseteq D_j} H_D$. The cost of the corresponding routing is $\sum_D |D|c(H_D(\mathcal{T}))$.

Our algorithm for the laminar demands case is an extension of the Goemans-Williamson primal-dual algorithm for the Steiner Forest Problem [13]. We begin by defining the primal and dual linear programs.

In the linear program below, the variable $x_{e,D}$ denotes whether $e \in H_D$. We denote by $\delta(S)$ the set of edges crossing a cut $S \subseteq V$, and by $\mathcal{S}_D$ the collection of cuts $S \subseteq V$ that separates a terminal set X_j with $D_j \supseteq D$. The cut constraints require that each terminal set X_j is connected by $\bigcup_{D \supseteq D_j} H_D$.

$$\begin{aligned} &\text{minimize} && \sum_{e, D \in \mathcal{D}} x_{e,D} \cdot |D| c_e && \\ &\text{subject to} && \sum_{D' \supseteq D} \sum_{e \in \delta(S)} x_{e,D'} \geq 1 && \forall D \in \mathcal{D}, S \in \mathcal{S}_D \end{aligned}$$

The corresponding dual linear program is as follows.

$$\begin{aligned} &\text{maximize} && \sum_{D \in \mathcal{D}, S \in \mathcal{S}_D} y_{D,S} && \\ &\text{subject to} && \sum_{D' \subseteq D} \sum_{S \in \mathcal{S}_{D'} : e \in \delta(S)} y_{D',S} \leq |D| c_e && \forall e, D \in \mathcal{D} \end{aligned}$$

3.1 Algorithm

The algorithm starts with a dual ascent stage in which it adds edges to forests $\{F_D\}_{D \in \mathcal{D}}$, and ends with a pruning stage. In the following discussion, for a demand set $D \in \mathcal{D}$ we say that $S \in \mathcal{S}_D$ is a *D-unsatisfied cut* if $(\bigcup_{D' \supseteq D} F_{D'}) \cap \delta(S) = \emptyset$. We also say that an edge e is *D-tight* if

$$\sum_{D' \subseteq D} \sum_{S \in \mathcal{S}_{D'} : e \in \delta(S)} y_{D',S} = |D| c_e.$$

In the dual ascent stage, the algorithm raises duals in phases, one per demand set $D \in \mathcal{D}$ in order of increasing size. In phase D, while there exists a D-unsatisfied cut it alternates between raising duals of the minimal D-unsatisfied cuts and adding D-tight edges to F_D. We say that S is an *active set* in the current iteration of the inner while loop if it is a minimal D-unsatisfied cut. The algorithm ensures that at the end of phase D, the edges F_D are paid for by the dual and F_D is a Steiner forest for terminal sets whose demand set contains D. In the pruning stage, the algorithm processes the demand sets in order of decreasing size and removes unnecessary edges from $\{F_D\}_{D\in\mathcal{D}}$ and returns $\{H_D\}_{D\in\mathcal{D}}$.

Algorithm 1 Primal-Dual Algorithm for Laminar Buy-at-Bulk

Initialize $F_D \leftarrow \emptyset$ for all $D \in \mathcal{D}$ and $y_{D,S} \leftarrow 0$ for all $D \in \mathcal{D}, S \subseteq V$.
(Dual ascent stage)
for $D \in \mathcal{D}$ in increasing order of size **do**
  (Start of phase D)
  while there exists a D-unsatisfied cut **do**
    Simultaneously raise $y_{D,S}$ for active sets S until some edge e goes D-tight.
    $F_D \leftarrow F_D + e$.
  end while
  (End of phase D)
end for
(End of dual ascent stage)
(Pruning stage)
$H_D \leftarrow F_D$ for all $D \in \mathcal{D}$.
for $D \in \mathcal{D}$ in decreasing order of size **do**
  for $e \in H_D$ **do**
    if $(H_D - e) \cup \bigcup_{D' \supsetneq D} H_{D'}$ is a Steiner forest for terminal sets with demand set D **then**
      $H_D \leftarrow H_D - e$.
    end if
  end for
end for
(End of pruning stage)
return $\{H_D\}_D$

The following lemma implies that we can efficiently find active sets.

▶ **Lemma 2.** *In any iteration in phase D, a set S is active if and only if it is a component of F_D and it separates a terminal set whose demand set contains D.*

Proof. Let S be an active set. By definition, S is a minimal cut in $\mathcal{S}_D$ such that $\bigcup_{D'\supseteq D} F_{D'} \cap \delta(S) = \emptyset$. Since $S \in \mathcal{S}_D$, it separates a terminal set whose demand set contains D. The algorithm processes the demand sets in increasing order of size, so we have $F_{D'} = \emptyset$ for $D' \supsetneq D$ and thus $F_D \cap \delta(S) = \emptyset$. This implies that $S \cap C = \emptyset$ or $S \cap C \supseteq C$ for every connected component C of F_D and so S is a superset of a union of connected components of F_D. By minimality, we have that S is a connected component of F_D.

For the converse, consider a connected component S' of F_D that separates a terminal set whose demand set contains D. By definition, we have $S' \in \mathcal{S}_D$. Since S' is a connected component of F_D and $F_{D'} = \emptyset$ for $D' \supsetneq D$, it is a minimal set in $\mathcal{S}_D$ such that $\bigcup_{D'\supseteq D} F_{D'} \cap \delta(S) = \emptyset$. Therefore S' is an active set. ◀

3.2 Analysis

Our analysis follows along the lines of the analysis for the Goemans-Williamson algorithm. We first establish that the primal and dual solutions generated by the algorithm are feasible.

▶ **Lemma 3.** *The primal solution* $\{H_D\}_{D\in\mathcal{D}}$ *and the dual solution* $\{y_{D,S}\}_{D\in\mathcal{D},S\subseteq V}$ *are feasible.*

Proof. We first prove that the primal solution is feasible. Consider an iteration during the pruning stage. We say that terminal set X_j is *H-disconnected* if it is disconnected with respect to edge set $\bigcup_{D\supseteq D_j} H_D$ and *H-connected* otherwise. We will show that all terminal sets are H-connected in all iterations of the pruning stage.

Observe that at the end of phase D, there are no D-unsatisfied cuts and $F_{D'} = \emptyset$ for $D' \supsetneq D$. Thus, all terminal sets with demand set D are connected with respect to edge set F_D. At the beginning of the pruning stage, we have $H_D = F_D$ for all $D \in \mathcal{D}$, and so all terminal sets are H-connected. Consider an iteration in which the algorithm deletes an edge e from H_D. By definition of H-disconnected, this can only cause a terminal set with demand set $D' \subseteq D$ to be H-disconnected. However, the algorithm will not delete e if it causes a terminal set with demand set D to be H-disconnected. Now consider a demand set $D' \subsetneq D$. Since $|D'| \leq |D|$, we still have $H_{D'} = F_{D'}$ so all terminal sets with demand set D' are H-connected. Thus, all terminal sets are H-connected throughout the pruning stage and so $\{H_D\}_{D\in\mathcal{D}}$ is a feasible primal solution.

The dual solution is feasible since the algorithm explicitly ensures that the dual variables in a tight constraint are not raised. ◀

Next, we show that in each phase D of the dual raising stage, the current active sets has average degree with respect to edges $\bigcup_{D'\supseteq D} H_{D'}$ (formally defined below) at most 2 in every iteration. This in turn implies that the primal solution has cost at most twice the total dual value. Since the dual is feasible, we have that the algorithm gives a 2-approximation. We bound the average degree of active sets by showing that $\bigcup_{D'\supseteq D} H_{D'}$ is a forest and that no inactive set has degree 1.

▶ **Lemma 4.** *For all* $D \in \mathcal{D}$*, we have that* $\bigcup_{D'\supseteq D} H_{D'}$ *is a forest.*

Proof. Suppose, towards a contradiction, that the statement is false. Let D be a maximal demand set such that $\bigcup_{D'\supseteq D} H_{D'}$ contains a cycle C. By maximality, there exists $e \in C \cap H_D$. Since e is in a cycle in $\bigcup_{D'\supseteq D} H_D$, we have that $(H_D - e) \cup \bigcup_{D'\supsetneq D} H_{D'}$ is still a Steiner forest for terminal sets with demand set D. Thus, the algorithm would have removed e from H_D and so we have a contradiction. ◀

For a subset of edges $E' \subseteq E$, let $\deg_{E'}(S) = |\delta(S) \cap E'|$ denote the number of edges in E' exiting S.

▶ **Lemma 5.** *Consider an iteration in phase D of the dual raising stage. Let S be a connected component of F_D in this iteration. If $S \notin \mathcal{S}_D$, then $\sum_{D'\supseteq D} \deg_{H_{D'}}(S) \neq 1$.*

*Proof of x*We prove the contrapositive. Suppose $\sum_{D'\supseteq D} \deg_{H_{D'}}(S) = 1$. Let e and $A \supseteq D$ be the unique edge and demand set, respectively, such that $e \in H_A \cap \delta(S)$. Since the algorithm did not delete e from H_A and $\bigcup_{D'\supseteq A} H_{D'}$ is acyclic by Lemma 4, there exists X_j with $D_j = A$ and $u, v \in X_j$ such that e is on the unique $u - v$ path in $\bigcup_{D'\supseteq A} H_{D'}$. Since $\sum_{D'\supseteq D} \deg_{H_{D'}}(S) = 1$, the path crosses S exactly once. Thus, we have that S separates u, v and so $S \in \mathcal{S}_A$. By definition of $\mathcal{S}_D$, we have $\mathcal{S}_A \subseteq \mathcal{S}_D$ and this completes the proof of the lemma. ◀

We are now ready to prove that the primal solution has cost at most twice the dual value.

▶ **Lemma 6.** $\sum_D \sum_{e \in H_D} |D| c_e \leq 2 \sum_{D,S} y_{D,S}$.

Proof. Using the fact that we only add tight edges, we have

$$\begin{aligned}
\sum_D \sum_{e \in H_D} |D| c_e &= \sum_D \sum_{e \in H_D} \left(\sum_{D' \subseteq D} \sum_{S \in \mathcal{S}_{D'} : e \in \delta(S)} y_{D',S} \right) \\
&= \sum_{D'} \sum_{S \in \mathcal{S}_{D'}} y_{D',S} \left(\sum_{D \supseteq D'} \sum_{e \in \delta(S) \cap H_D} 1 \right) \\
&= \sum_{D'} \sum_{S \in \mathcal{S}_{D'}} y_{D',S} \left(\sum_{D \supseteq D'} \deg_{H_D}(S) \right) \\
&= \sum_{D'} \sum_{S \in \mathcal{S}_{D'}} y_{D',S} \deg_{\bigcup_{D \supseteq D'} H_D}(S).
\end{aligned}$$

The second equality is obtained by rearranging, and the last follows from the fact that each edge is in H_D for at most one $D \supseteq D'$.

Suppose that in an iteration in phase D', the dual for each active set is raised by Δ. This implies $\sum_{S \in \mathcal{S}_{D'}} y_{D',S} \deg_{\bigcup_{D \supseteq D'} H_D}(S)$ increases by $\Delta \cdot \sum_{S \text{ active}} \deg_{\bigcup_{D \supseteq D'} H_D}(S)$, and $\sum_{D,S} y_{D,S}$ increases by $\Delta \cdot \#$ active sets. So it suffices to prove that in each phase D' and in each iteration within the phase, the average degree of active sets is at most 2:

$$\sum_{S \text{ active}} \deg_{\bigcup_{D \supseteq D'} H_D}(S) \leq 2 \cdot \# \text{ active sets}.$$

Fix an iteration in phase D'. Note that each active set corresponds to some connected component of $F_{D'}$ by Lemma 2. Let G' be a graph whose nodes are connected components of $F_{D'}$ and whose edge set is $\bigcup_{D \supseteq D'} H_D$. The degree of a node in G' is equal to the degree of the corresponding set with respect to edge set $\bigcup_{D \supseteq D'} H_D$. Let us say that a node of G' corresponding to an active set is an *active node*, and that any other node is *inactive*. We want to show that the average degree of active nodes in G' is at most 2. Suppose we remove all isolated nodes from G'. In the resulting graph, by Lemma 5 the degree of each inactive node is at least 2, and by Lemma 4 the average degree is at most 2. So the claim follows. ◀

Lemmas 3 and 6 gives us the following theorem.

▶ **Theorem 7.** *Algorithm 1 is a 2-approximation for network design with coverage costs in the laminar demands setting.*

4 A logarithmic approximation for the sunflower demands setting

We now consider the sunflower demands setting. First we note that a polylogarithmic hardness of approximation follows by an approximation-preserving reduction from a special case of the buy-at-bulk problem called the *single-cable buy-at-bulk problem*.

In the single-cable buy-at-bulk problem, we are given a graph $G = (V, E)$ with costs c_e on edges, a load function $f(x) = L + x$ if $x > 0$ and $f(0) = 0$, and g terminal pairs (s_i, t_i). The goal is to find routes R_i for each terminal pair minimizing the cost $\sum_e f(|\{i : R_i \ni e\}|) \cdot c_e$.

Andrews and Zhang [2] showed that the single-cable buy-at-bulk problem has no $O(\log^{\frac{1}{4}-\gamma} g)$-approximation for any constant γ under standard complexity-theoretic assumptions.[3]

Observe that we can reinterpret any instance of this problem as an instance of network design with sunflower demands over the same graph as follows: let P denote a set of L "common" packets, and for each i, P_i denote a unique singleton packet; for each terminal pair (s_i, t_i) we have a group $X_i = \{s_i, t_i\}$ with demand $P \cup P_i$. Then any solution for the former problem is also a solution for the latter with the same cost and vice versa. Thus, we get the following hardness result.

▶ **Theorem 8.** *Network design with coverage costs in the sunflower demands setting does not admit an $O(\log^{\frac{1}{4}-\gamma} g)$-approximation for any constant γ unless* NP $\subseteq$ ZPTIME$(n^{\text{polylog}\, n})$.

Next we prove the main technical result of this section which says that we can find a group spanner of linear size with stretch $O(\log g)$.

▶ **Lemma 9.** *Given an unweighted graph $G = (V, E)$ ($c_e = 1$ for all $e \in G$) and terminal sets $X_1, \ldots, X_g$ such that $V = \bigcup_j X_j$, we can construct in polynomial time a $(14, 4 \log g)$ group spanner.*

Before we prove Lemma 9, we observe that, together with Lemma 1, it implies the following result for unweighted instances of the sunflower demands setting with vertex set $V = \bigcup_j X_j$.

▶ **Theorem 10.** *Network design with coverage costs in the sunflower demands setting admits an $O(\log g)$ approximation over unweighted graphs with vertex set $V = \bigcup_j X_j$.*

In the remainder of the section we will focus on unweighted graphs and write $|H|$ to denote the cost (i.e., the number of edges) of subgraph H. Let us recall some notation: for a subgraph H, $\text{St}_H(X)$ denotes the cost of an optimal (i.e., minimum cost) Steiner tree over vertex set X in H, and $d_H(u, v)$ denotes the distance between vertices u, v in H. Let T denote a minimum spanning tree of the given graph G.

Now we prove Lemma 9. To that end we consider *uniform* group spanner instances where the following holds for all j: for all strict subsets S of X_j, there exists an edge $(x, y) \in E$ such that $x \in S, y \in X_j \setminus S$. In other words, there exists an optimal Steiner tree for each X_j with no Steiner vertices and it is easy to find.

Next we show that in order to establish Lemma 9 it suffices to solve uniform instances. We can transform any given group spanner instance over an unweighted graph G with $V = \bigcup_j X_j$ into a uniform instance as follows: add to X_j all Steiner vertices in the 2-approximate Steiner tree given by the MST heuristic [31] applied over X_j in G and let X'_j be the resulting set. Since X'_j is the set of all vertices of a Steiner tree, the group spanner instance with terminal sets $X'_1, \ldots, X'_g$ is a uniform one.

Say we obtain subgraph H after solving the above uniform instance and H satisfies $\text{St}_H(X'_j) \leq \beta\, \text{St}_G(X'_j)$ for all j and $|H| \leq \alpha|T|$. We show that H is in fact a $(2\alpha, 2\beta)$ group spanner for the original instance. The MST heuristic guarantees that $\text{St}_G(X'_j) \leq 2\, \text{St}_G(X_j)$; which implies $\text{St}_H(X_j) \leq 2\beta\, \text{St}_G(X_j)$. Finally, let F^* denote an optimal Steiner forest for $X_1, \ldots, X_g$ in G. In an unweighted instance, we have that $|F^*| \geq |T|/2$. This is because $V = \bigcup_j X_j$ and each component of the forest has at least one edge[4] so $|F^*| \geq |V|/2 \geq |T|/2$. Since, $|H| \leq \alpha|T|$ we get the cost guarantee, $|H| \leq 2\alpha|F^*|$.

[3] While [2] considers a different problem, it was remarked in [3] that the construction can be used for single-cable buy-at-bulk.

[4] We assume without loss of generality that $|X_j| \geq 2$ for all j

This implies that to prove Lemma 9 we only need to solve uniform group spanner instances. In the remainder of this section, we focus on uniform instances and for ease of exposition write X_j in place of X'_j.

▶ **Lemma 11.** *Given any uniform group spanner instance with terminal sets X_j, there exists a subset of edges A of size $|A| \leq 6|T|$ such that for $H := A \cup T$ we have $\mathrm{St}_H(X_j) \leq (2\log g)\,\mathrm{St}_G(X_j)$ for all j.*

Since $|H| = |A| + |T| \leq 7|T|$ and $\mathrm{St}_H(X_j) \leq (2\log g)\,\mathrm{St}_G(X_j)$, we get that H is a $(14, 4\log g)$ group spanner that satisfies the desired bounds in Lemma 9.

We now move on to present a constructive proof of Lemma 11. We assume that terminals of X_j are ordered $x_{j,1}, x_{j,2}, \ldots$ such that for $i > 1$, there exists an edge $(x_{j,i}, x_{j,k}) \in E$ for some $k < i$; we call this edge a *satisfying edge* for $x_{j,i}$. For ease of notation, we drop the indices when they do not matter and write (x, y) to denote x's satisfying edge. Note that such an ordering always exists, e.g. a preordering of the (uniform) Steiner tree over X_j with any root. We say that a terminal $x_{j,i} \in X_j$ is *unsatisfied*[5] in a spanning subgraph H if $d_H(x_{j,i}, \{x_{j,1} \ldots, x_{j,i-1}\}) > 2\log g$. Note that a single vertex may correspond to multiple satisfied/unsatisfied terminals of different groups. The following fact implies that subgraphs in which all terminals are satisfied are group spanners with $\beta = 2\log g$.

▶ Fact 1. If H is a spanning subgraph such that $d_H(x_{j,i}, \{x_{j,1} \ldots, x_{j,i-1}\}) \leq 2\log g$ for all $i > 1$, then there exists a Steiner tree for X_j in H with total size at most $(2\log g)\,\mathrm{St}_G(X_j)$.

Our algorithm starts with the MST T and adds satisfying edges to it in order to construct H. In order to bound the cost of these edges, the algorithm maintains an arc set E' defined over the vertex set V. Let G' denote the directed graph (V, E'). At the beginning of the algorithm, E' is empty. We use *arcs* to refer to directed edges in E' and simply *edges* for edges in E. Our algorithm works in two phases. In the first phase, for each unsatisfied terminal, the algorithm adds its satisfying edge only if we can add an oriented copy of it to E' and modify nearby arcs in E' such that the out-degree of every node is at most 2. The main lemma is that the number of unsatisfied terminals at the end of this phase is at most $|V|$, and so we can simply add their satisfying edges in the second phase. We use the following notation for the algorithm: $\delta^+(x)$ denotes the number of edges of E' that are oriented away from x; $\Gamma(x) \subseteq V$ denotes the set of terminals reachable from x via a directed path in E' of length at most $\log g$.

At the end of the algorithm every vertex is satisfied. Fact 1 then implies that $H = T \cup A_1 \cup A_2$ is a group spanner with $\beta = 2\log g$. So we only need to bound the sizes of A_1 and A_2. Since there is a one-to-one correspondence between edges in A_1 and arcs in E', the following lemma implies that $|A_1| = |E'| \leq 2|V|$.

▶ **Lemma 12.** *We have $\delta^+(x) \leq 2$ for all $x \in V$.*

Proof. We prove the lemma by induction on the iterations of the algorithm. The base case ($E' = \emptyset$) is trivial. The interesting case is when $\delta^+(x) = 2$ at the beginning of the iteration and the algorithm adds (x, y) to E' oriented from x to y. At this point, we have $\delta^+(x) = 3$, $\delta^+(z) \leq 1$ and all other terminals on the $x - z$ path have out-degree at most 2 by the inductive hypothesis. When the algorithm flips the arcs on the path, it decrements $\delta^+(x)$ by 1, increments $\delta^+(z)$ by 1 and does not affect the out-degrees of other terminals on the path. This proves the lemma. ◀

[5] We define the lowest indexed vertex $x_{j,1}$ to be always satisfied.

Algorithm 2 Algorithm for uniform graph spanner instances

(Phase 1)
$E', A_1, A_2 \leftarrow \emptyset$
while there exists x that is unsatisfied in $T \cup A_1$ and $z \in \Gamma(x)$ such that $\delta^+(z) \leq 1$ **do**
  Add x's satisfying edge (x, y) to E' oriented from x to y
  Add (x, y) to A_1
  if $\delta^+(x) > 2$ **then**
    Flip directions of arcs in G' along $x - z$ path
  end if
end while
(Phase 2)
For every x unsatisfied in $T \cup A_1$, add its satisfying edge (x, y) to A_2
return $A = A_1 \cup A_2$

Next we bound $|A_2|$.

▶ **Lemma 13.** $|A_2| \leq |V|$.

Proof. First we prove that, even if we ignore edge directions, the length of the smallest cycle (i.e. girth) in E' is at least $\log g$. Assume, towards a contradiction, that there is an undirected cycle of length $k \leq \log g$ in E'. Let (x, y) be the last arc added in the cycle. Before the algorithm added it, there is a path from x to y of length $k - 1$ in A corresponding to the other arcs in the cycle. This contradicts the condition for adding (x, y); in particular, x is not unsatisfied.

Let $U = \{x_{j,i} : x_{j,i} \text{ unsatisfied in } T \cup A_1\}$. For $x_{j,i} \in U$, we have $\delta^+(z) = 2$ for all $z \in \Gamma(x_{j,i})$ since otherwise we would have added its satisfying edge in phase 1. Since the girth of E' is at least $\log g$, we have a full binary tree of depth $\log g$ rooted at $x_{j,i}$ in E'. This implies $|\Gamma(x_{j,i})| \geq g$. Furthermore, for any $x_{j,i}, x_{j,k} \in U$ with $i > k$, we have $\Gamma(x_{j,i}) \cap \Gamma(x_{j,k}) = \emptyset$ because otherwise $d_{T \cup A_1}(x_{j,i}, x_{j,k}) \leq 2 \log g$ and $x_{j,i}$ would not have been unsatisfied in $T \cup A_1$. Therefore any terminal can belong to at most one $\Gamma(x_{j,i})$ per j, giving us $\sum_{x_{j,i} \in U} |\Gamma(x_{j,i})| \leq g|V|$. Hence we get the desired bound: $|V| \geq \sum_{x_{j,i} \in U} |\Gamma(x_{j,i})|/g \geq g|U|/g = |U| = |A_2|$. ◀

Lemmas 12 and 13 imply that $|A_1| + |A_2| \leq 3|V|$. Furthermore, the algorithm ensures that all the terminals are satisfied in $T \cup A_1 \cup A_2$. Together with Fact 1, we get Lemma 11.

References

1 Ashok Anand, Vyas Sekar, and Aditya Akella. SmartRE: an architecture for coordinated network-wide redundancy elimination. In *ACM SIGCOMM*, 2009.
2 M. Andrews and L. Zhang. Bounds on fiber minimization in optical networks with fixed fiber capacity. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 1, pages 409–419 vol. 1, March 2005.
3 Matthew Andrews. Hardness of buy-at-bulk network design. In *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 115–124. IEEE, 2004.
4 Matthew Andrews and Lisa Zhang. Approximation algorithms for access network design. *Algorithmica*, 34(2):197–215, 2002.
5 Baruch Awerbuch and Yossi Azar. Buy-at-bulk network design. In *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science*, pages 542–547, 1997.

6 Baruch Awerbuch, Alan Baratz, and David Peleg. Cost-sensitive analysis of communication protocols. In *Proceedings of the ninth annual ACM symposium on Principles of distributed computing*, PODC'90, pages 177–187, New York, NY, USA, 1990. ACM.

7 Siddharth Barman and Shuchi Chawla. Traffic-redundancy aware network design. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'12, pages 1487–1498. SIAM, 2012.

8 Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, FOCS'96, pages 184–193, Washington, DC, USA, 1996. IEEE Computer Society.

9 BlueCoat: WAN Optimization. `http://www.bluecoat.com`.

10 Michael Elkin and David Peleg. $(1+\epsilon,\beta)$-spanner constructions for general graphs. *SIAM Journal on Computing*, 33(3):608–631, 2004.

11 Paul Erdős. Extremal problems in graph theory. In *Theory of Graphs and its Applications (Proc. Sympos. Smolenice, 1963)*, pages 29–36. Publ. House Czechoslovak Acad. Sci., Prague, 1964.

12 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 448–455, 2003.

13 M. Goemans and D. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

14 S. Guha, A. Meyerson, and K. Munagala. Hierarchical placement and network design problems. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 603–612, 2000.

15 Anupam Gupta, Amit Kumar, Martin Pál, and Tim Roughgarden. Approximation via cost-sharing: A simple approximation algorithm for the multicommodity rent-or-buy problem. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'03, pages 606–617, 2003.

16 Anupam Gupta, Amit Kumar, and Tim Roughgarden. Simpler and better approximation algorithms for network design. In *STOC'03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 365–372, 2003.

17 A. Hayrapetyan, C. Swamy, and É. 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 Samir Khuller, Balaji Raghavachari, and Neal Young. Balancing minimum spanning and shortest path trees. In *Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms*, SODA'93, pages 243–250, Philadelphia, PA, USA, 1993. Society for Industrial and Applied Mathematics.

19 Amit Kumar, Anupam Gupta, and Tim Roughgarden. A constant-factor approximation algorithm for the multicommodity rent-or-buy problem. In *Proceedings of the 43rd Symposium on Foundations of Computer Science*, FOCS'02, pages 333–344, 2002.

20 F Thomson Leighton and Ankur Moitra. Extensions and limits to vertex sparsification. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 47–56. ACM, 2010.

21 Adam Meyerson, Kamesh Munagala, and Serge Plotkin. Cost-distance: Two metric network design. *SIAM J. Comput.*, 38(4):1648–1659, December 2008.

22 Ankur Moitra. Approximation algorithms for multicommodity-type problems with guarantees independent of the graph size. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 3–12. IEEE, 2009.

23 Seth Pettie. Low distortion spanners. In *Automata, Languages and Programming*, pages 78–89. Springer, 2007.

24 Riverbed Networks: WAN Optimization. `http://www.riverbed.com/us/solutions/optimization`.

25 David B. Shmoys, Chaitanya Swamy, and Retsef Levi. Facility location with service installation costs. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1088–1097, 2004.

26 Neil T. Spring and David Wetherall. A protocol-independent technique for eliminating redundant network traffic. In *Proceedings of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communication – SIGCOMM'00*, pages 87–95, Stockholm, Sweden, 2000.

27 Zoya Svitkina and Éva Tardos. Facility location with hierarchical facility costs. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 153–161, 2006.

28 Kunal Talwar. The Single-Sink Buy-at-Bulk LP Has Constant Integrality Gap. In *Proceedings of the 9th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 475–486, 2002.

29 Mikkel Thorup and Uri Zwick. Approximate distance oracles. *J. ACM*, 52(1):1–24, January 2005.

30 Mikkel Thorup and Uri Zwick. Spanners and emulators with sublinear distance errors. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 802–809. ACM, 2006.

31 Vijay V Vazirani. *Approximation Algorithms*. Springer, 2001.

Online Set Cover with Set Requests*

Kshipra Bhawalkar[1], Sreenivas Gollapudi[2], and Debmalya Panigrahi[3]

1 **Google Inc., Mountain View, CA, U.S.**
`kshipra@google.com`
2 **Microsoft Research Search Labs, Mountain View, CA, U.S.**
`sreenig@microsoft.com`
3 **Duke University, Durham, NC, U.S.**
`debmalya@cs.duke.edu`

Abstract

We consider a generic online allocation problem that generalizes the classical online set cover framework by considering requests comprising a set of elements rather than a single element. This problem has multiple applications in cloud computing, crowd sourcing, facility planning, etc. Formally, it is an online covering problem where each online step comprises an offline covering problem. In addition, the covering sets are capacitated, leading to packing constraints. We give a randomized algorithm for this problem that has a nearly tight competitive ratio in both objectives: overall cost and maximum capacity violation. Our main technical tool is an online algorithm for packing/covering LPs with nested constraints, which may be of interest in other applications as well.

1998 ACM Subject Classification F.2.2 Non-numerical Algorithms and Problems

Keywords and phrases Online Algorithms, Set Cover

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.64

1 Introduction

In recent years, significant research has been conducted in online allocation problems (see [1] and [8] for a comprehensive discussion on online algorithms), often motivated by inherently online modern applications such as internet advertising, crowd sourcing, scheduling in the cloud, etc. We continue this research effort in this paper by considering a generic allocation problem that is motivated by various real-world applications and generalizes the well-studied online set cover framework. In the online set cover problem [2], a collection of subsets (of given costs) of a universe of elements are given offline and elements from the universe arrive online. At any time, the algorithm must maintain a monotonically increasing (over time) collection of subsets of minimum cost that cover all the elements that have arrived thus far. In the capacitated version, every set also has a given capacity which represents the maximum number of elements it can cover. In this paper, we consider a natural generalization of this problem, where instead of a single new element, a subset of elements arrives in each online step. Note that this generalization is meaningful only in the capacitated situation since the elements arriving in the same online step use up only one unit of capacity of the covering sets. In the uncapacitated (i.e., infinite capacity) scenario, the elements arriving in a single step can be thought of as arriving sequentially.

* Part of this work was done when all the authors were at Microsoft Research.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 64–79

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

To formally describe our problem, we need to introduce some notation and terminology. Departing from the usual set cover notation, we think of every element as a resource and every covering set as a facility that provides some subset of resources. This ties the notation to natural applications of the problem and helps us distinguish between the request sets (that arrive online) and the covering sets (that are offline and called facilities now). Let U be the set of n different resources (such as goods and services) and $\mathbf{S} = \{S_1, S_2, \ldots, S_m\}$ be a set of facilities, each of which can provide some subset $S_j \subseteq U$ of resources. Each facility S_j also has an associated cost c_j and capacity t_j. The above are given offline. There are k *requests* that arrive online. In each online step, a request $R_i \subseteq U$ arrives, and has to be satisfied by assigning a subset of facilities to it that can cumulatively provide all the resources requested, i.e. by using a subset of facilities $\mathbf{T}_i \subseteq \mathbf{S}$ such that $R_i \subseteq \cup_{T \in \mathbf{T}_i} T$. The capacity of a facility is the maximum number of requests it can serve, and the ratio of the number of requests served by a facility to its capacity is called its *congestion*. The goal is to minimize the sum of costs of the facilities purchased by the algorithm. We call this the Cover-SetReq problem. Our focus, in this paper, will be to design an online algorithm for the Cover-SetReq problem.

Our work was motivated by various applications of the above general framework in emerging domains. We give a couple of motivating examples below:

- **Distributed Computing:** In distributed computing environments such as cloud computing and crowd sourcing, each computing unit (e.g., a human or a server) provides a subset of computing resources and has a maximum capacity. The goal is to minimize cost while allocating each arriving task to a subset of computing units that have adequate resources to solve it.
- **Facility Planning:** The goal is to minimize the cost of facilities (each of which can provide a subset of services and has a maximum capacity) to serve service requests that grow over time as new customers are added.
- **Subscription Markets.** In addition to traditional products, the internet has emerged as the principal medium for the sale of services based on information and data management including access to data sets and computing resources (see, e.g., [7, 13, 14]). Examples include the Windows Azure Marketplace[1], Amazon Web Services[2], etc. These services are typically sold as *subscriptions* comprising one or more resources that come as a bundle with an usage limit. The consumer objective is to satisfy their data/computing needs which arrive over time at minimum cost by buying an optimal set of subscriptions.

Our main result is a polynomial-time online algorithm for the Cover-SetReq problem. To state its competitive ratio, let us use an equivalent (up to a constant factor in the competitive ratio, by a standard doubling search approach) description of the Cover-SetReq problem, where in addition to the input described above, a cost bound $\mathbf{C}$ is given offline with the guarantee that there exists a feasible solution, i.e. a solution that does not use more than the capacity of any facility and has total cost at most $\mathbf{C}$. Then, an online algorithm for the Cover-SetReq problem is said to have a bi-criteria competitive ratio of (α, β) if its total cost is at most $\alpha\mathbf{C}$ and for every facility, the number of requests that it is used to satisfy is at most β times its capacity (i.e., its congestion is at most β). Our main theorem obtains poly-logarithmic factors for both α and β.

[1] http://datamarket.azure.com
[2] http://aws.amazon.com

$$\sum_{j:j\in[m]} c_j x_j \le \mathbf{C} \tag{1}$$
$$\sum_{j:u\in S_j} y_{ij} \ge 1 \qquad \forall u \in R_i, \forall i \in [k] \tag{2}$$
$$y_{ij} \le 2x_j \qquad \forall i \in [k], \forall j \in [m] \tag{3}$$
$$\sum_{i:i\in[k]} y_{ij} \le x_j t_j \qquad \forall j \in [m] \tag{4}$$
$$0 \le y_{ij} \le 1 \qquad \forall i \in [k], \forall j \in [m] \tag{5}$$
$$0 \le x_j \le 1 \qquad \forall j \in [m] \tag{6}$$

Figure 1 Linear program for the Cover-SetReq problem.

▶ **Theorem 1.** *There is a randomized online algorithm for the* Cover-SetReq *problem that has a competitive ratio of* (α, β) *where* $\alpha = O(\log(mn)\log(kmn))$ *and* $\beta = O(\log n(\log m + \log\log n)\log(kmn))$.

We note that this theorem is nearly tight since there are logarithmic lower bounds for both α and β: (1) there is a (randomized) lower bound of $\Omega(\log m \log n)$ [2, 12] for the competitive ratio of the online set cover problem, which holds for the cost objective of the Cover-SetReq problem, and (2) there is a lower bound of $O(\log m)$ [3] for the online restricted assignment problem, which holds for the congestion objective of the Cover-SetReq problem.

We remark that some applications motivate a version of the Cover-SetReq problem with soft capacities, i.e. where multiple copies of a facility can be used in the solution. Clearly, our algorithm has a poly-logarithmic competitive ratio for this problem as well. However, this problem can be solved using an alternative (simpler) technique:

- linearize the cost of all copies of a facility other than its first copy by losing a factor of 2 (see Jain and Vazirani [11])
- reduce the mixed LP to a covering LP of exponential size (but with an efficient separation oracle) by eliminating precedence packing constraints
- obtain a fractional solution to the covering problem using a standard template given by Buchbinder and Naor [9] (see also Gupta and Nagarajan [10])
- obtain an integer solution using our randomized rounding procedure.

The details of these steps appear in the appendix. The second step fails for the Cover-SetReq problem, i.e., when we have hard capacities.

Our Techniques. First, we define an LP for the Cover-SetReq problem (in Figure 1). Let x_j denote whether facility S_j is opened and y_{ij} indicate whether facility S_j is used to serve request R_i. We enforce that each resource in every request is served (i.e. $\sum_{j:u\in S_j} y_{ij} \ge 1$) and to ensure a bounded integrality gap, that $y_{ij} \le 2x_j$ (the factor 2 is for technical reasons). In addition, the total cost is bounded ($\sum_{j\in[m]} c_j \le \mathbf{C}$) and the congestion on every facility S_j is bounded ($\sum_{i\in[n]} y_{ij} \le x_j t_j$).

As mentioned earlier, we will obtain a fractional solution to this LP (which will violate some of the constraints) using combinatorial techniques and then round this fractional solution online. This recipe was suggested originally by Alon *et al.* [2] for the online set cover problem and has since been used extensively for online algorithms (see the survey by Buchbinder and Naor [9] for more details). The online rounding algorithm is the easier of the two steps and (somewhat delicately) combines rounding techniques for the online [5] and offline [16] set cover problems.

Obtaining a fractional algorithm turns out to be much more challenging. Since the Cover-SetReq problem is represented by a mixed packing covering LP, following Azar *et al.* [5], for each request, we use a sequence of multiplicative updates on a prefix of facilities with $x_j < 1$ ordered by a carefully chosen function that represents the derivative of the overall potential of the solution. However, unlike in [5], since requests contain multiple resources, the prefix is not unique, rather it depends on the resource being considered. Moreover, it is not immediate as to how we can compare between two facilities, one with many resources but higher cost and another with fewer resources but lower cost. To complicate matters further, at any stage of the multiplicative weights update process, different resources are at various stages of being served: some have been completely served, some only partially served, while others have not been served at all. The resources that have been fully served should cease to influence the ordering since the facilities providing these resources are no longer contributing to serving these resources.

Since each online step is an offline set cover problem, we inherit its greedy property and order facilities by the potential increase *per resource* that each facility provides (call it the *scaled cost*). To address the issue of some resources having already been completely served, we make these prefix orderings *dynamic*: once a resource has been completely served, it is not included in defining scaled costs thereafter. Moreover, since each resource only appears in some of the facilities, we introduce the notion of a *resource specific* prefix ordering, which is a subsequence of the overall prefix ordering.

For the fully open facilities (i.e., $x_j = 1$), we need to ensure that the maximum congestion is small. For this purpose, we follow a technique introduced by Aspnes *et al* [3] (see also [6, 4]) for online load balancing, where a greedy algorithm on an exponential potential function of the machine loads is used. Our main technical contribution is a procedure that co-ordinates between the greedy selection of facilities in prefixes, multiplicative weight updates on these multiple prefixes, and greedy assignment of requests to facilities according to an exponential potential function of their congestion for fully open facilities.

Roadmap. The next section presents the online algorithm, whose competitive ratio is derived in two parts: the analysis of the fractional solution is in section 3 and the analysis of the randomized rounding procedure to convert the fractional solution into an integer one is in section 4. In the appendix, we present a simpler algorithm for the soft-capacitated version of the Cover-SetReq problem (section A).

2 Description of the Algorithm

The algorithm has three phases: (a) an offline pre-processing phase, (b) an online phase that produces a fractional solution, and (c) an online rounding phase that produces an integer solution from the fractional solution. The last two phases are interleaved. Recall that we are given a bound on the cost $\mathbf{C}$ and the number of requests k in advance. Let OPT denote a solution that has congestion at most 1 on every facility and total cost at most $\mathbf{C}$.

The Offline Pre-processing Phase. First, we discard all facilities S_j with $c_j > \mathbf{C}$ from $\mathbf{S}$. Clearly, none of these facilities were being used by OPT. From now on, m will denote the size of $\mathbf{S}$ after this step. Next, we divide the cost of each facility by $\frac{\mathbf{C}}{m}$. After this scaling, the total cost of OPT is at most m. For any facility $S_j \in \mathbf{S}$, if $c_j < \frac{1}{k}$, we increase c_j to $\frac{1}{k}$. After this transformation, the total cost of OPT is at most $\left(1 + \frac{1}{k}\right) m < 2m$.

Let $x_j^{(i)}$ denote the value of variable x_j at the end of the updates for request R_i. Note that the non-decreasing property of x_j requires that $x_j^{(i)} \geq x_j^{(i-1)}$. We say that facility S_j is *fully open* if $x_j = 1$, and *partially open* otherwise. We initialize x_j to $x_j^{(0)} = \frac{1}{m}$ for all facilities $S_j \in \mathbf{S}$. Therefore, initially, all facilities are partially open.

Online Updates to the Fractional Solution. Suppose a new request R_i arrives online. Any resource $u \in R_i$ is said to be *satisfied* if $\sum_{j:u \in S_j} y_{ij} \geq 1$. Clearly, R_i is satisfied when all resources in R_i are satisfied. We start by setting $x_j^{(i)} = x_j^{(i-1)}$ (required by monotonicity of the fractional solution). We increase the value of $x_j^{(i)}$ on selected facilities S_j in small increments over multiple *rounds* and make corresponding increments in y_{ij}. Each round, in turn, consists of multiple *iterations*.

Let $\overline{R}_i$ denote the set of resources in R_i that are not yet satisfied at the beginning of the round, i.e., $\overline{R}_i = \{u \in R_i : \sum_{j:u \in S_j} y_{ij} < 1\}$. The increments in the values of $x_j^{(i)}$ and y_{ij} in any particular round are based on defining a sequence of facilities containing u (called the *prefix* for u and denoted $\mathbf{P}_i(u)$) for each individual resource $u \in \overline{R}_i$. For some of the resources $u \in \overline{R}_i$, we will also define an additional facility in $\mathbf{S} \setminus \mathbf{P}_i(u)$ as the *boundary facility* for u, and denote the index of this facility by $p_i(u)$. Let $\widehat{\mathbf{P}}_i(u) = \mathbf{P}_i(u) \cup S_{p_i(u)}$; we call this the *closed prefix* of u.

To describe the update rule of the fractional variables and the construction of the prefixes, we need some additional notation. For every facility S_j, we partition requests into those that arrive before S_j is fully open (denote this set $R_0(j)$) and those that arrive after (denote this set $R_1(j)$). For the request that was being served when the facility became fully open, we consider the part of the request that arrived while $x_j < 1$ in $R_0(j)$ and the rest of the request in $R_1(j)$. The *virtual congestion* (denoted $\tilde{L}_j$) of a facility S_j is defined as

$$\tilde{L}_j = \begin{cases} x_j & \text{if } x_j < 1 \\ 1 + \sum_{i:R_i \in R_1(j)} \frac{y_{ij}}{t_j} & \text{if } x_j = 1. \end{cases}$$

Now, we define a function (A is a constant that we will fix later)

$$\psi_j = \begin{cases} \frac{c_j}{t_j} & \text{if } x_j < 1 \\ \frac{c_j A^{\tilde{L}_j}(A-1)}{t_j} & \text{if } x_j = 1. \end{cases}$$

The updates for all facilities in prefix $\mathbf{P}_i(u)$ and the boundary facility $S_{p_i(u)}$ are collectively called an iteration for resource u, and the iterations for all resources in $\overline{R}_i$ constitute a round for request R_i. The update rule for a round is given in Algorithm 1, where N is a discretization parameter that we set to kmn^2. One important point to note is that if a partially open facility S_j belongs to k_j closed prefixes, then the value of $x_j^{(i)}$ increases in multiplicative update steps k_j times in a single round.

Definition of the Prefixes. We initialize the prefix $\mathbf{P}_i(u)$ to the empty sequence for every resource $u \in \overline{R}_i$. The prefixes are populated in a sequence of steps, where in each step, we add a carefully selected facility to some of the prefixes. To describe a step, we need some additional notation. Let $\overline{\overline{R}}_i$ denote the set of resources in $\overline{R}_i$ whose prefix has not been fully defined yet. Clearly, $\overline{\overline{R}}_i$ equals $\overline{R}_i$ at the beginning of a round. Further, let $\mathbf{S}(i)$ denote the collection of facilities in $\mathbf{S}$ that overlap $\overline{\overline{R}}_i$ and have not been used in a previous step (i.e. is not part of any prefix currently). Initially, $\mathbf{S}(i) = \{S_j \in \mathbf{S} : S_j \cap \overline{R}_i \neq \emptyset\}$.

For any facility $S_j \in \mathbf{S}(i)$, let its *scaled cost* be $\phi_j = \frac{\psi_j}{|S_j \cap \overline{\overline{R}}_i|}$.

Algorithm 1 A Single Round of the Fractional Algorithm

- $\overline{R}_i = \{u \in R_i : \sum_{j:u \in S_j} y_{ij} < 1\}$.
- Create closed prefixes $\widehat{\mathbf{P}}_i(u)$ simultaneously for all resources $u \in \overline{R}_i$.
- For every facility S_j: initialize $\Delta x_j = \Delta y_{ij} = 0$.
- For every resource $u \in \overline{R}_i$: for every partially open facility $S_j \in \widehat{\mathbf{P}}_i(u)$, we increase Δx_j by $\frac{x_j^{(i)}}{c_j N}$ (sequentially, in arbitrary order over the closed prefixes $\widehat{\mathbf{P}}_i(u)$ for all resources $u \in \overline{R}_i$).
- For every facility S_j: if S_j is partially open, we set $\Delta y_{ij} = \min\left((\Delta x_j)t_j, 2(x_j^{(i)} + \Delta x_j) - y_{ij}\right)$; if S_j is fully open, we set $\Delta y_{ij} = \frac{1}{\psi_j N}$.
- For every facility S_j: increase $x_j^{(i)}$ by Δx_j and y_{ij} by Δy_{ij}.

In each step, the algorithm performs the following operations:

1. Find facility $S_j \in \mathbf{S}(i)$ that has the least value of ϕ_j; let us denote its index by j^*.
2. Remove S_{j^*} from $\mathbf{S}(i)$.
3. Let $\mathbf{x}(u) = \sum_{j:S_j \in \mathbf{P}_i(u)} x_j^{(i)} + x_{j^*}^{(i)}$. For each resource $u \in S_{j^*} \cap \overline{\overline{R}}_i$, if $\mathbf{x}(u) < 1$, then we add S_{j^*} to the prefix $\mathbf{P}_i(u)$. Otherwise, if $\mathbf{x}(u) \geq 1$, then we define S_{j^*} as the boundary facility for resource u, i.e., $p_i(u) = j^*$ and remove u from $\overline{\overline{R}}_i$.
4. Re-define $\mathbf{S}(i)$ (since $\overline{\overline{R}}_i$ might have changed) and re-compute ϕ_j for all facilities $S_j \in \mathbf{S}(i)$ (even if a facility continues to be in $\mathbf{S}(i)$, its scaled cost might have changed since $\overline{\overline{R}}_i$ has changed).

Note that it might so happen that for a resource $u \in R_i$, even after including all facilities containing u in the prefix $\mathbf{P}_i(u)$, $\sum_{j:S_j \in \mathbf{P}_i(u)} x_j^{(i)} < 1$. In this case, the boundary facility for u is undefined, and its closed prefix is identical to its prefix.

Online Randomized Rounding. There are two decisions that the integer algorithm must make on receiving a new request R_i. First, it needs to decide which set of facilities it wants to open. Since decisions are irrevocable in the online model, the open facilities form a monotonically growing set over time. Next, the algorithm must decide which of the open facilities it will use to satisfy request R_i. As we describe below, both these decisions are made by the integer algorithm based on the fractional solution that it maintains using the algorithm given above.

To simplify the analysis later, we will consider two copies of each facility: a *blue* copy and a *red* copy. Note that this is without loss of generality, up to a constant factor loss in the competitive ratio for both the cost and the congestion. First, we define a randomized process that controls the opening of blue copies of facilities in the integer algorithm. Let $\mathbf{S}_o(i)$ denote the set of facilities whose blue copies are open after request R_i has been satisfied, and $X_j^{(i)}$ be an indicator random variable whose value is 1 if facility $i \in \mathbf{S}_o(i)$ and 0 otherwise. Let $x_j^{(i)}$ be the value of variable x_j in the fractional solution after request R_i has been completely assigned (fractionally). For a parameter $\alpha = \Theta(\log(kmn))$, the integer algorithm maintains the following invariant for every facility S_j and request R_i:

$$\mathbb{P}[X_j^{(i)} = 1] = \min(\alpha \cdot x_j^{(i)}, 1), \tag{7}$$

using the rule for opening facilities in Algorithm 2. Next, we need to use the open facilities to satisfy request R_i. Let Y_{ij} be the indicator variable for facility S_j being used to serve

request R_i. Define

$$z_{ij} = \begin{cases} 0 & \text{if } X_j^{(i)} = 0 \\ \frac{y_{ij}}{2x_j^{(i)}} & \text{if } X_j^{(i)} = 1 \text{ and } x_j^{(i)} < \frac{1}{\alpha} \\ \alpha \cdot y_{ij} & \text{otherwise.} \end{cases}$$

The assignment rule for request R_i is given in Algorithm 2.

Algorithm 2 Satisfying a Single Request R_i in the Integer Algorithm

Opening Facilities:

- For every facility S_j whose blue copy is not already open, open it with probability $\min\left(\frac{\alpha(x_j^{(i)} - x_j^{(i-1)})}{1-\alpha \cdot x_j^{(i-1)}}, 1\right)$. (Eqn. 7 is satisfied by this rule using conditional probabilities.)

Satisfying Request R_i:

- For every open facility S_j, we set $Y_{ij} = 1$ independently with probability z_{ij}.
- For every resource $u \in R_i$ such that no facility containing u was selected in the previous step, set $Y_{ij} = 1$ for the red copy of *any* facility S_j such that $u \in S_j$, after opening the facility if necessary.

3 Analysis of the Fractional Algorithm

We note that the fractional solution maintains the invariant $\sum_{R_i \in R_0(j)} \frac{y_{ij}}{t_j} \leq x_j$ for every facility S_j. This invariant ensures that the actual congestion of any facility is always at most its virtual congestion (denoted $\tilde{L}_j$; see section 2 for its formal definition). Therefore, it suffices to bound the total cost and the maximum virtual congestion on the facilities. For this purpose, we design a potential function that combines these two objectives: $\gamma_j = c_j x_j A^{\tilde{L}_j / x_j}$ for some $A \in (1, 2)$ that we will fix later. Note that we can rewrite the potential function as

$$\gamma_j = \begin{cases} A c_j x_j & \text{if } x_j < 1 \\ c_j A^{\tilde{L}_j} & \text{if } x_j = 1. \end{cases}$$

The potential function is continuous and monotonically non-decreasing. We define the overall potential $\Gamma = \sum_{j : S_j \in \mathbf{S}} \gamma_j$.

The next lemma bounds the potential function at the end of the pre-processing step.

▶ **Lemma 2.** *At the end of the pre-processing step,* $\Gamma \leq m$.

Proof. There are m partially open facilities, the cost of each of which is at most m. Since we initialize $x_j^{(0)} = 1/m$ for all the m facilities, the lemma follows. ◀

Next we will bound the increase in potential due to online updates to the fractional solution. Recall that for any request R_i, there are several rounds, each comprising multiple iterations, one for every resource in $\overline{R}_i$. Our general plan is the following: we will first bound the increase in potential in a *single iteration* and then bound the total number of iterations performed by the algorithm (overall, for all requests and for all rounds corresponding to a request).

Increase in Potential in a Single Iteration. First, note that a facility S_j might belong to multiple closed prefixes in a single round. Therefore, the value of x_j for partially open facilities S_j and that of $\tilde{L}_j$ for fully open facilities changes from one iteration to another in the same round. To reconcile this inconsistency, we bound the increase of these variables in a single round in the next lemma.

▶ **Lemma 3.** *For any partially open facility S_j, the value of $x_j^{(i)}$ can increase by a multiplicative factor of at most e in a single round. Similarly, for any fully open facility S_j, the value of $A^{\tilde{L}_j}$ can increase by a multiplicative factor of at most 2 in a single round.*

Proof. First, consider a partially open facility S_j. Since there are at most n iterations in a round, the multiplicative factor by which the value of $x_j^{(i)}$ increases in a single round is at most

$$\left(1+\frac{1}{Nc_j}\right)^n \leq \left(1+\frac{k}{N}\right)^n \leq e,$$

where the first inequality follows from the fact that $c_j \geq \frac{1}{k}$ for all facilities S_j and the second inequality holds since $N \geq nk$.

Next, consider a fully open facility S_j with virtual congestion $\tilde{L}_j$ at the beginning of the round. The multiplicative factor by which $A^{\tilde{L}_j}$ increases in a single round is at most

$$A^{\frac{\Delta y_{ij}}{t_j}} - 1 = (1+(A-1))^{\frac{\Delta y_{ij}}{t_j}} - 1 \leq 2(A-1)\frac{\Delta y_{ij}}{t_j} \leq \frac{2(A-1)n}{c_j A^{\tilde{L}_j}(A-1)N} \leq \frac{2nk}{AN} \leq 2,$$

where the first inequality uses the fact that for any $y \geq x \geq 0$,

$$\left(1+\frac{1}{x}\right)^{1/y} \leq e^{x/y} \leq 1+\frac{2x}{y} \tag{8}$$

(we call this *local linearization*); the second inequality holds since virtual congestion, and therefore ψ_j, is non-decreasing and there are at most n iterations in a round; the third inequality uses $c_j \geq \frac{1}{k}$ for all facilities S_j and $\tilde{L}_j \geq 1$ for any fully open facility S_j; and the last inequality follows from $N \geq nk$. ◀

The next lemma bounds the increase in potential of the fractional solution in a single iteration.

▶ **Lemma 4.** *The increase in potential in a single iteration for any resource $u \in \overline{R}_i$ is at most $\frac{10A}{N}$.*

Proof. Note that the increase in potential in an iteration can be attributed to two possible sources: increase in cost for partially open facilities in the closed prefix $\widehat{\mathbf{P}}_i(u)$ and increase in virtual congestion of the boundary facility $S_{p_i(u)}$.

First, we bound the increase in cost. Recall that at the beginning of the round,

$$\sum_{j:S_j \in \widehat{\mathbf{P}}_i(u)} x_j^{(i)} = \left(\sum_{j:S_j \in \mathbf{P}_i(u)} x_j^{(i)}\right) + x_{p_i(u)}^{(i)} \leq 1+1 = 2.$$

However, the value of $x_j^{(i)}$ increases over the various iterations in the round, and therefore, it is possible that $\sum_{j:S_j \in \widehat{\mathbf{P}}_i(u)} x_j^{(i)} > 2$ at the beginning of the iteration for resource u. Nevertheless, by Lemma 3, we can claim that $\sum_{j:S_j \in \widehat{\mathbf{P}}_i(u)} x_j^{(i)} \leq 2e < 6$.

The increase in potential due to increments in $x_j^{(i)}$ for all partially open facilities $S_j \in \widehat{\mathbf{P}}_i(u)$ is

$$A \sum_{j:S_j \in \widehat{\mathbf{P}}_i(u)} \frac{c_j x_j^{(i)}}{N c_j} = \frac{A}{N} \sum_{j:S_j \in \widehat{\mathbf{P}}_i(u)} x_j^{(i)} < \frac{6A}{N},$$

where the inequality follows from the observation above.

Next, we consider the increase in potential due to the increase in virtual congestion of facility $S_{p_i(u)}$, if it is fully open. If the virtual congestion before the iteration was $\tilde{L}_{p_i(u)}$, then the increase in potential is

$$c_{p_i(u)} A^{\tilde{L}_{p_i(u)}} \left(A^{\frac{\Delta y_{ip_i(u)}}{t_{p_i(u)}}} - 1 \right) \leq 2 \frac{\psi_{p_i(u)} t_{p_i(u)}}{(A-1)} \cdot \frac{2(A-1)\Delta y_{ip_i(u)}}{t_{p_i(u)}} = \frac{4}{N} < \frac{4A}{N},$$

where the first inequality uses local linearization (see Eqn. 8) and Lemma 3. ◀

Total Number of Iterations. Recall that OPT is a a feasible integer solution with cost at most $\mathbf{C}$ and congestion at most 1 on each facility. Let $\text{OPT}(R_i)$ denote the facilities used by OPT to satisfy request R_i. An iteration for resource $u \in \overline{R}_i$ is in one of the following two categories:

1. At least one facility in $\text{OPT}(R_i)$ is in the prefix $\mathbf{P}_i(u)$.
2. No facility in $\text{OPT}(R_i)$ is in the prefix $\mathbf{P}_i(u)$.

The number of iteration of the first category is bounded by the next lemma.

▶ **Lemma 5.** *The total number of iterations of the first category is $O(Nm \log m)$.*

Proof. Let S_{j^*} be a facility in OPT. The number of iterations where S_{j^*} is in the prefix is $O(Nc_{j^*} \log m)$ since:

- x_{j^*} is initialized to $\frac{1}{m}$ in the pre-processing phase.
- $x_{j^*} < 1$ before the last round where x_{j^*} increases. Therefore, by Lemma 3, $x_{j^*} < e$ at the end of the round.
- x_{j^*} increases by a multiplicative factor of $\left(1 + \frac{1}{Nc_{j^*}}\right)$ in every iteration where it belongs to the prefix.

The lemma follows by summing over all facilities $S_{j^*} \in \text{OPT}$. ◀

Now, we focus on iterations of the second category. We partition rounds into ones where $\overline{R}_i$ changes (we call these *dynamic* rounds) and ones where $\overline{R}_i$ does not change (we call these *static* rounds). The number of iterations in dynamic rounds is bounded by the next lemma.

▶ **Lemma 6.** *The total number of iterations in dynamic rounds is $O(N)$.*

Proof. Since $\overline{R}_i$ changes, i.e., loses a resource in any dynamic round, a single request R_i can have at most $|R_i| \leq n$ dynamic rounds. Since there are at most n iterations in each round and at most k requests overall, the lemma follows from $N > kn^2$. ◀

Now, we focus on counting the number of iterations of the second category in static rounds. Recall that for any partially open facility S_j, we set $y_{ij} = \min(2x_j^{(i)}, t_j(x_j^{(i)} - x_j^{(i-1)}))$ at the end of the round. Let $\overline{T}$ be the collection of partially open facilities such that $y_{ij} = 2x_j^{(i)}$ and $T = \mathbf{S} \setminus \overline{T}$ be all the remaining facilities. The next lemma lower bounds the contribution of facilities in T in any iteration of the second category in a static round.

▶ **Lemma 7.** *For any static round and any iteration of the second category for resource $u \in \overline{R}_i$, it holds that*

$$\sum_{j:S_j \in T \cap \widehat{\mathbf{P}}_i(u)} x_j^{(i)} \geq 1/2.$$

Proof. Suppose for some resource u, $\sum_{j:S_j \in T \cap \widehat{\mathbf{P}}_i(u)} x_j^{(i)} < 1/2$. Note that since the iteration for u is of the second category, the boundary facility must be defined and $\sum_{j \in \widehat{\mathbf{P}}_i(u)} x_j^{(i)} \geq 1$. We conclude that $\sum_{j \in \overline{T} \cap \widehat{\mathbf{P}}_i(u)} x_j^{(i)} \geq 1/2$. For every facility $S_j \in \overline{T}$, $y_{ij} = 2x_j^{(i)}$. However, in that case, $\sum_{j \in \overline{T} \cap \widehat{\mathbf{P}}_i(u)} y_{ij} \geq 1$ and the resource u is satisfied at the end of this round. In other words, the round is dynamic, which is a contradiction. ◀

For a resource u, we refer to $\sum_{j:u \in S_j} y_{ij}$ as its *coverage*. We will show in the next lemma that the increase in coverage on resources in $\overline{R}_i$, averaged over the iterations of the second category in a static round, is large. Before stating the lemma, we need to introduce some notation. For any facility S_j, let us partition $\overline{R}_i \cap S_j$ as follows: U_j contains resources that have S_j in their prefix, V_j contains resources that have S_j as the boundary facility, and W_j contains the rest of the resources. Note that prefixes of resources in W_j are filled first, followed by those in V_j, and finally those in U_j.

Now, consider a facility $S_{j^*} \in \text{OPT}(R_i)$. Let $\overline{B}$ be the set of resources in $S_{j^*} \cap \overline{R}_i$ that have iterations of the second category; let $\overline{b} = |\overline{B}|$.

▶ **Lemma 8.** *The total increase in coverage on resources of $\overline{B}$ in a single static round is at least $\frac{\overline{b} \cdot |S_{j^*} \cap \overline{R}_i|}{2N\psi_{j^*}}$.*

Proof. Let $u \in \overline{B}$ and D_u denote the set of resources of $S_{j^*} \cap \overline{R}_i$ whose closed prefixes were filled with or after $\widehat{\mathbf{P}}_i(u)$. Clearly, $D_u \subseteq \overline{\overline{R}}_i$ for all steps of constructing the prefix till the step that filled $\widehat{\mathbf{P}}_i(u)$. Since S_{j^*} was not inserted in $\mathbf{P}_i(u)$, therefore the scaled cost ϕ_j for every facility S_j in the closed prefix of u satisfies

$$\phi_j = \frac{\psi_j}{|U_j \cup V_j|} \leq \frac{\psi_{j^*}}{|D_u|}.$$

For any such facility $S_j \in T \cap \widehat{\mathbf{P}}_i(u)$, the total increase of y_{ij} is at least $\frac{x_j |U_j \cup V_j|}{N\psi_j} \geq \frac{x_j |D_u|}{N\psi_{j^*}}$. Summing over all such facilities S_j and using Lemma 7, we can conclude that the total increase in coverage on u is at least $\frac{|D_u|}{2N\psi_{j^*}}$.

Now, let us order the resources $u \in \overline{B}$ in which $\widehat{\mathbf{P}}_i(u)$ got filled. For the first u in this order, $D_u = S_{j^*} \cap \overline{R}_i$. Each subsequent D_u loses precisely one resource, the one whose prefix was just filled. For the last u in in the order, $D_u = U_{j^*} \cup \{u\}$. Note that the iterations for resources in U_{j^*} are in the first category. Thus, $\overline{B} \subseteq V_{j^*} \cup W_{j^*}$. Let $|U_{j^*}| = p$ and $|V_{j^*} \cup W_{j^*}| = q$. Then,

$$|U_{j^*} \cup V_{j^*} \cup W_{j^*}| = |S_{j^*} \cap \overline{R}_i| = p + q.$$

Adding up the increases in coverage obtained from the above expression,

$$\sum_{u \in \overline{B}} \frac{|D_u|}{2N\psi_{j^*}} = \sum_{i=p+1}^{p+q} \frac{i}{2N\psi_{j^*}} \geq \frac{p(p+q)}{2N\psi_{j^*}} = \frac{\overline{b} \cdot |S_{j^*} \cap \overline{R}_i|}{2N\psi_{j^*}}.$$

◀

We now consider two subcases:

- (a) static rounds where S_{j^*} is partially open, and
- (b) static rounds where S_{j^*} is fully open.

The advantage with subcase (a) is that the value of ψ_{j^*} in the previous lemma depends only on the facility S_{j^*} and not on the state of the algorithm.

▶ **Lemma 9.** *Let S_{j^*} be a facility in $\text{OPT}(R_i)$. Then, the number of iterations of the second category in static rounds for resources in $S_{j^*} \cap R_i$, where S_{j^*} is partially open, is $2\left(\frac{c_{j^*}}{t_{j^*}}\right) N \ln n$.*

Proof. Let $z_{j^*} = \sum_{u \in S_{j^*} \cap R_i} \max\left(1 - \sum_{j:u \in S_j} y_{ij}, 0\right)$. By Lemma 8, the decrease in z_{j^*} in any static round comprising $\overline{b}$ iterations is at least $\frac{\overline{b} \cdot |S_{j^*} \cap \overline{R}_i|}{2N(c_{j^*}/t_{j^*})}$. Since z_{j^*} decreases from at most n to 0, it follows that the total number of iterations is

$$\int_{z_{j^*}=n}^{0} 2N \cdot \frac{c_{j^*}}{t_{j^*}} \cdot \frac{dz_{j^*}}{z_{j^*}} = 2N\left(\frac{c_{j^*}}{t_{j^*}}\right) \ln n. \qquad \blacktriangleleft$$

The next corollary follows by summing over all requests R_i and facilities S_{j^*} in OPT.

▶ **Corollary 10.** *The total number of iterations of the second category for resources u in static rounds for requests R_i such that there exists a partially open facility S_{j^*} that is used to satisfy R_i in OPT and contains u is at most $O(Nm \log n)$.*

We are left with subcase (b), i.e., when facility S_{j^*} is fully open. Let $\mathbf{L}_{j^*}$ be the virtual congestion on facility S_{j^*} at the end of the algorithm. Then, at any intermediate stage of the algorithm when S_{j^*} was fully open,

$$\psi_{j^*} \leq \frac{c_{j^*} A^{\mathbf{L}_{j^*}} (A-1)}{t_{j^*}}.$$

Using the same logic as Lemma 9, we obtain the next lemma.

▶ **Lemma 11.** *Let S_{j^*} be a facility in $\text{OPT}(R_i)$. Then, the number of iterations of the second category in static rounds for resources in $S_{j^*} \cap R_i$, where S_{j^*} is fully open, is $2\left(\frac{c_{j^*} A^{\mathbf{L}_{j^*}}(A-1)}{t_{j^*}}\right) N \ln n$.*

Now, note that the congestion on S_{j^*} in OPT is at most 1, i.e. the number of requests served by facility S_{j^*} is at most t_{j^*}. Therefore, we obtain the next corollary.

▶ **Corollary 12.** *The total number of iterations of the second category for resources u in static rounds for requests R_i such that there exists a fully open facility S_{j^*} that is used to satisfy R_i in OPT and contains u is at most $2N \ln n(A-1) \sum_{j^*:S_{j^*} \in \text{OPT}} c_{j^*} A^{\mathbf{L}_{j^*}}$.*

Now, we add up all the bounds that we have obtained on the increase of the potential to obtain the next lemma.

▶ **Lemma 13.** *At the end of the algorithm, the final potential $\Gamma_f = O(m \log(mn))$.*

Proof. By summing up over the individual bounds on the number of iterations in the various categories,

$$\Gamma_f = O(m \log(mn)) + 20A(\ln n)(A-1) \sum_{j:S_j \in \text{OPT}} c_j A^{\mathbf{L}_{j^*}} \leq O(m \log(mn)) + \frac{1}{2}\Gamma_f,$$

by choosing $A = 1 + \frac{1}{80 \ln n}$. The lemma follows. $\blacktriangleleft$

The next corollary follows from the definition of the potential function Γ.

▶ **Corollary 14.** *The total cost of the fractional solution is $O(m \log(mn))$ and the maximum congestion on a facility is $O(\log n(\log m + \log\log n))$.*

4 Analysis of Online Randomized Rounding

Recall that the fractional solution maintains the following invariant for any facility S_j and any request R_i:

$$y_{ij} \leq 2x_j^{(i)}. \tag{9}$$

First, we consider red copies of facilities.

▶ **Lemma 15.** *The probability that the red copy of any facility is opened is at most $e^{-\Omega(\alpha)}$.*

Proof. We first consider the scenario where for a resource u, no facility $S_j \in \mathbf{S}(u)$ is opened in the integer solution, i.e., $\sum_{j:u\in S_j} X_j^{(i)} = 0$. Since $y_{ij} \leq 2x_j^{(i)}$ (Eqn. 9) and $\sum_{j:S_j\in\mathbf{S}(u)} y_{ij} \geq 1$, it follows that $\sum_{j:u\in S_j} \alpha x_j^{(i)} \geq \frac{\alpha}{2}$. Therefore, the probability that $\sum_{j:u\in S_j} X_j^{(i)} = 0$ is at most $\prod_{j:u\in S_j}\left(1-\alpha x_j^{(i)}\right) = e^{-\Omega(\alpha)}$ by Eqn. 7.

Next, consider the scenario where $\sum_{j:u\in S_j} X_j^{(i)} \geq 1$ but $\sum_{j:u\in S_j} Y_{ij} = 0$, i.e., even though facilities that contain resource u are open, none of them have been assigned to request R_i. Let $\mathbf{A}_i$ and $\mathbf{B}_i$ respectively denote the set of facilities S_j with $x_j^{(i)} < \frac{1}{\alpha}$ and those with $x_j^{(i)} \geq \frac{1}{\alpha}$. Clearly, all facilities in $\mathbf{B}_i$ are open in the integer solution and some subset of facilities in $\mathbf{A}_i$ is open. We consider two subcases. First, suppose $\sum_{j:S_j\in\mathbf{B}_i, u\in S_j} y_{ij} \geq 1/2$. Then,

$$\sum_{j:S_j\in\mathbf{B}_i,u\in S_j} = \sum_{j:S_j\in\mathbf{B}_i\cap\mathbf{S}(u)} \alpha y_{ij} \geq \frac{\alpha}{2}.$$

Therefore, the probability of $\sum_{j:u\in S_j} Y_{ij} = 0$ is at most $\prod_{j:S_j\in\mathbf{B}_i,u\in S_j}(1-z_{ij}) = e^{-\Omega(\alpha)}$.

Finally, suppose $\sum_{j:S_j\in\mathbf{B}_i,u\in S_j} y_{ij} < 1/2$. Then, $\sum_{j:S_j\in\mathbf{A}_i\cap\mathbf{S}(u)} y_{ij} \geq 1/2$.

In this case, we first estimate the expectation and bound the probability of deviation of random variables z_{ij}. We have

$$\mathbb{E}\left[\sum_{j:S_j\in\mathbf{A}_i,u\in S_j} z_{ij}\right] = \sum_{j:S_j\in\mathbf{A}_i,u\in S_j}\left(\frac{y_{ij}}{2x_j^{(i)}}\right)\mathbb{P}\left[X_j^{(i)}=1\right]$$

$$= \sum_{j:S_j\in\mathbf{A}_i,u\in S_j}\left(\frac{y_{ij}}{2x_j^{(i)}}\right)\alpha x_j^{(i)} = \sum_{j:S_j\in\mathbf{A}_i,u\in S_j}\frac{y_{ij}\alpha}{2} \geq \sum_{j:S_j\in\mathbf{A}_i,u\in S_j}\frac{\alpha}{4}.$$

Since $y_{ij} \leq 2x_j^{(i)}$, we can use Chernoff bounds (see, e.g., [15]) to claim that with probability $1-e^{-\Omega(\alpha)}$,

$$\sum_{j:S_j\in\mathbf{A}_i\cap\mathbf{S}(u)} z_{ij} = \Omega(\alpha). \tag{10}$$

On the other hand, if Eqn. 10 holds, then the probability of $\sum_{j:u\in S_j} Y_{ij} = 0$ is

$$\prod_{j:S_j\in\mathbf{A}_i,u\in S_j}(1-z_{ij}) = e^{-\Omega(\alpha)}.$$ ◀

We choose $\alpha = \Theta(\log(knm))$ and use linearity of expectation over all requests and resources to conclude that the red copies of the facilities can be ignored by incurring an additive $O(1)$ loss in the approximation ratio.

We will now bound the expected cost and congestion of the blue copies of facilities.

▶ **Lemma 16.** *The expected total cost of blue copies of facilities in the integer solution is at most α times the cost of the fractional solution.*

Proof. The proof is an immediate consequence of Eqn. 7 using linearity of expectation. ◀

▶ **Lemma 17.** *With probability $1 - o(1)$, the congestion on every facility in the integer solution is $O(\alpha)$ times their virtual congestion in the fractional solution.*

Proof. We split the congestion on a facility S_j in the integer solution into its congestion from requests in $R_0(j)$ (before S_j is fully open in the fractional solution) and $R_1(j)$ (after S_j is fully open in the fractional solution). By linearity of expectation, the expected congestion due to requests in $R_1(j)$ is at most $\alpha \sum_{i:R_i \in R_1(j)} \frac{y_{ij}}{t_j}$.

On the other hand, the expected congestion due to requests in $R_0(j)$ is at most

$$\sum_{i:R_i \in R_0(j)} \frac{y_{ij}}{2x_j^{(i)} t_j} \leq \sum_{i:R_i \in R_0(j)} \frac{x_j^{(i)} - x_j^{(i-1)}}{2x_j^{(i)}} \leq \int_{1/m}^{1} \frac{dw}{w} = \ln m = O(\alpha),$$

where the last bound follows from the choice of $\alpha = \Theta(\log knm)$. ◀

Using standard techniques (bounding the maximum possible congestion if the above lemma fails, and therefore obtaining a bound on its contribution to the expectation), we can convert the high probability bound on the maximum congestion in the above lemma to the same bound (up to constants) on the expectation of the maximum congestion.

5 Conclusion and Future Work

We have given an algorithm for a generic online covering problem where each individual request comprises a set of elements. The competitive ratio of our algorithm is poly-logarithmic in the input parameters. While such dependence on the number of elements and subsets in the set system is matched by existing lower bounds, it is not clear whether our dependence on the number of requests is necessary. We leave the resolution of this dependence as an open question. Our problem represents a nesting of online and offline covering problems. An intriguing open problem is to obtain a formal algorithmic framework for packing/covering LPs that are revealed online in stages where each stage is an offline packing/covering LP.

Acknowledgement. We thank an anonymous reviewer for suggesting the alternative (and simpler) technique for the Cover-SetReq problem with soft capacities. D. Panigrahi is supported in part by startup funds from Duke University.

References

1 Susanne Albers. Online algorithms: a survey. *Math. Program.*, 97(1-2):3–26, 2003.
2 Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. The online set cover problem. *SIAM J. Comput.*, 39(2):361–370, 2009.
3 James Aspnes, Yossi Azar, Amos Fiat, Serge A. Plotkin, and Orli Waarts. On-line routing of virtual circuits with applications to load balancing and machine scheduling. *J. ACM*, 44(3):486–504, 1997.

4 Yossi Azar. On-line load balancing. In *Online Algorithms*, pages 178–195, 1996.
5 Yossi Azar, Umang Bhaskar, Lisa K. Fleischer, and Debmalya Panigrahi. Online mixed packing and covering. In *SODA*, 2013.
6 Yossi Azar, Joseph Naor, and Raphael Rom. The competitiveness of on-line assignments. *J. Algorithms*, 18(2):221–237, 1995.
7 M. Balazinska, B. Howe, and D. Suciu. Data markets in the cloud: An opportunity for the database community. *PVLDB*, 4(12):1482–1485, 2011.
8 Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, New York, NY, USA, 1998.
9 Niv Buchbinder and Joseph Naor. The design of competitive online algorithms via a primal-dual approach. *Foundations and Trends in Theoretical Computer Science*, 3(2-3):93–263, 2009.
10 Anupam Gupta and Viswanath Nagarajan. Approximating sparse covering integer programs online. In *ICALP (1)*, pages 436–448, 2012.
11 Kamal Jain and Vijay V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *J. ACM*, 48(2):274–296, 2001.
12 Simon Korman. On the use of randomization in the online set cover problem. *M.S. thesis, Weizmann Institute of Science*, 2005.
13 Paraschos Koutris, Prasang Upadhyaya, Magdalena Balazinska, Bill Howe, and Dan Suciu. Querymarket demonstration: Pricing for online data markets. *PVLDB*, 5(12):1962–1965, 2012.
14 Chao Li and Gerome Miklau. Pricing aggregate queries in a data marketplace. In *WebDB*, pages 19–24, 2012.
15 Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1997.
16 Prabhakar Raghavan and Clark D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

A A simpler algorithm for the Cover-SetReq problem with soft capacities

Here we describe a simpler algorithm for the Cover-SetReq problem with soft capacities that follows from previous work. The algorithm follows by reducing the linear program for the Cover-SetReq problem with soft capacities which has mixed packing and covering constraints to one with just covering constraints. An online solution for covering program can be constructed using existing techniques [9, 10].

Note that the integer programming formulation of the Cover-SetReq problem with soft capacities is as follows:

$$\textbf{(P1)}\ \text{Minimize} \sum_{j=1}^{m} c_j x_j \qquad \text{subject to}$$

$$\sum_{j:u\in S_j} y_{ij} \geq 1 \quad \forall i \in [k], u \in R_i$$

$$y_{ij} \leq x_j \quad \forall i \in [k], j \in [m]$$

$$\sum_{i=1}^{k} y_{ij} \leq x_j t_j \qquad \forall j \in [m]$$

$$y_{ij} \in \{0,1\}, x_j \in \mathbb{N} \quad \forall i \in [k], j \in [m]$$

First we will show that the same problem can be solved using the following formulation while losing only a constant factor of 2 in the objective. This is based on an observation for Jain and Vazirani [11] along with some further ideas to obtain a covering LP.

$$\textbf{(P2)}\ \text{Minimize}\ \sum_{j=1}^{m} c_j x_j + \sum_{i=1}^{k}\sum_{j=1}^{m} \frac{c_j}{t_j}\cdot y_{ij} \qquad \text{subject to}$$

$$\sum_{j\in J} x_j + \sum_{j\in\{j:u\in S_j\}\setminus J} y_{ij} \geq 1 \quad \forall i\in[k], u\in R_i, J\subseteq\{j:u\in S_j\}$$

$$y_{ij}\geq 0, x_j\geq 0 \qquad \forall i\in[k], j\in[m]$$

▶ **Lemma 18.** *The program (P2) is a linear relaxation of (P1) with a factor 2 loss in objective. In particular,*

- $\text{OPT}(P2)\leq 2\cdot\text{OPT}(P1)$ *where* $\text{OPT}(P)$ *denotes the value of the optimal feasible solution.*
- *Any feasible solution* (x',y') *of (P2) can be mapped to a solution of the program (P1) with the same value of the objective provided it satisfies* $y_{ij}\leq x_j$ *for all* $1\leq i\leq k, 1\leq j\leq m$.

Proof. Consider the optimal feasible solution (x,y) of (P1). Define (x',y') as follows:

$$x'_j = min\{1,x_j\} \qquad \forall j\in[m]$$
$$y'_{ij} = y_{ij} \quad \forall j\in[m], i\in[k]$$

First we will show that $y'_{ij}\leq x'_j$. Since (x,y) is an optimal feasible solution, $y_{ij}\leq 1$. Then by the definition of x'_j, $y'_{ij}=y_{ij}\leq min\{1,x_j\}=x'_j$. It then follows that the first constraint in the LP (P2) holds since y_{ij} satisfy the first inequality in the program (P1). Next we bound the objective. Trivially, $\sum_{j=1}^{m} c_j x'_j \leq \sum_{j=1}^{m} c_j x_j = \text{OPT}(P1)$. Finally,

$$\sum_{j=1}^{m}\sum_{i=1}^{k}\frac{c_j}{t_j}y'_{ij} = \sum_{j=1}^{m} c_j \sum_{i=1}^{k} y_{ij}/t_j \leq \sum_{j=1}^{m} c_j x_j = \text{OPT}(P2).$$

It then follows that

$$\text{OPT}(P1)\leq \sum_{j=1}^{m} c_j x'_j + \sum_{j=1}^{m}\sum_{i=1}^{k}\frac{c_j}{t_j}y_{ij} \leq 2\text{OPT}(P1).$$

Next consider a feasible solution (x',y') of (P2) with $y_{ij}\leq x_j$ for all $1\leq i\leq k, 1\leq j\leq m$. Construct a solution (x,y) of (P1) as follows:

$$\forall 1\leq j\leq m \quad x_j = x'_j + \textstyle\sum_{i=1}^{k}\frac{y'_{ij}}{t_j}$$
$$\forall 1\leq j\leq m, 1\leq i\leq k \qquad y_{ij} = y'_{ij}$$

The first constraint of (P1) is obviously true. Since $y'_{ij}\leq x'_j$, it follows that $y_{ij}\leq x'_j\leq x_j$. Moreover,

$$\sum_{i=1}^{k} y_{ij}/t_j = \sum_{i=1}^{k} y'_{ij}/t_j \leq x_j.$$

Finally,

$$\sum_{j=1}^{m} c_j x_j = \sum_{j=1}^{m} c_j\left(x'_j + \sum_{i=1}^{k} y'_{ij}\right) = \sum_{j=1}^{m} c_j x'_j + \sum_{j=1}^{m}\sum_{k=1}^{n}\frac{c_j}{t_j}x_j.$$

◀

Finally note that the requirement $y'_{ij} \leq x'_j$ on a solution (x', y') of the program (P2) is without loss of generality. Any solution that violates this constraint can be fixed by lowering the value of y'_{ij} to x'_j. This still maintains all of the constraints while lowering the objective value.

We have thus obtained a covering linear program, any solution to which can be mapped to a feasible solution of the original IP with only a factor 2 loss in the objective. It is possible to construct a solution to program (P2) in an online manner using the techniques of Buchbinder and Naor [9]. (See also Gupta and Nagarajan [10].) The resulting solution can then be rounded using our randomized rounding procedure.

Lowest Degree k-Spanner: Approximation and Hardness

Eden Chlamtáč[1] and Michael Dinitz[2]

1 **Ben Gurion University, IL**
chlamtac@cs.bgu.ac.il
2 **Johns Hopkins University, U.S.**
mdinitz@cs.jhu.edu

Abstract

A k-spanner is a subgraph in which distances are approximately preserved, up to some given stretch factor k. We focus on the following problem: Given a graph and a value k, can we find a k-spanner that minimizes the maximum degree? While reasonably strong bounds are known for some spanner problems, they almost all involve minimizing the total number of edges. Switching the objective to the degree introduces significant new challenges, and currently the only known approximation bound is an $\tilde{O}(\Delta^{3-2\sqrt{2}})$-approximation for the special case when $k = 2$ [Chlamtáč, Dinitz, Krauthgamer FOCS 2012] (where Δ is the maximum degree in the input graph). In this paper we give the first non-trivial algorithm and polynomial-factor hardness of approximation for the case of general k. Specifically, we give an LP-based $\tilde{O}(\Delta^{(1-1/k)^2})$-approximation and prove that it is hard to approximate the optimum to within $\Delta^{\Omega(1/k)}$ when the graph is undirected, and to within $\Delta^{\Omega(1)}$ when it is directed.

1998 ACM Subject Classification G.2.2 Graph Theory: Graph algorithms

Keywords and phrases Graph spanners, approximation algorithms, hardness of approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.80

1 Introduction

A spanner of a graph is a sparse subgraph that approximately preserves distances. Formally, a k-spanner of a graph $G = (V, E)$ is a subgraph H of G in which $d_H(u, v) \leq k \cdot d_G(u, v)$ for all $u, v \in V$, where d_H and d_G denote shortest path distances in H and G, respectively[1]. Graph spanners were originally introduced in the context of distributed computing [22, 23], and since then have been extensively studied from both a distributed and a centralized perspective. Much of this work has focused on the fundamental tradeoffs between stretch, size, and total weight, such as the seminal result of Althöfer et al. that every graph admits a $(2k-1)$-spanner with at most $n^{1+1/k}$ edges [1] and its many extensions (e.g. to dealing with total weight [7]). Spanners have also appeared as fundamental building blocks in a wide range of applications, from routing in computer networks [25] to property testing of functions [4].

In parallel with this work on the fundamental tradeoffs there has been a line of work on approximating spanners. In this setting we are usually given an input graph G and a stretch value k, and our goal is to construct the best possible k-spanner. If "best" is measured in terms of the total number of edges, then clearly the construction of [1] gives

[1] Equivalently, a subgraph H is a k-spanner if $d_H(u, v) \leq k$ for every edge (u, v) in G.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 80–95

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

an $O(n^{2/(k+1)})$-approximation (for odd k), simply because $\Omega(n)$ is a trivial lower bound on the size of any spanner of a connected graph. However, when the objective function is to minimize the maximum degree, there are no non-trivial fundamental bounds like there are for the number of edges, so it is natural to consider the optimization problem. Moreover, degree objectives are notoriously difficult (consider degree-bounded minimum spanning trees [24] as opposed to general minimum spanning trees), and so almost all work on approximation algorithms for spanners has focused on minimizing the number of edges, as opposed to maximum degree.

We call the problem of minimizing the degree of a k-spanner the Lowest Degree k-Spanner problem (which we will abbreviate LDkS). For directed graphs, the degree is the sum of the in- and out-degrees.[2] Kortsarz and Peleg initiated the study of the maximum degree of a spanner, giving an $O(\Delta^{1/4})$-approximation for LD2S [21] (where Δ is the maximum degree of the input graph). This was only recently improved to $\tilde{O}(\Delta^{3-2\sqrt{2}+\epsilon})$ for arbitrarily small $\epsilon > 0$ by Chlamtáč, Dinitz, and Krauthgamer [10]. The only known hardness for LD2S was $\Omega(\log n)$ [21]. Despite the length of time since minimizing the degree was first considered (over 15 years) and the significant amount of work on other spanner problems, no nontrivial upper or lower bounds were known previous to this work for LDkS when $k \geq 3$.

1.1 Our results and techniques

We give the first nontrivial upper and lower bounds for the approximability of Lowest Degree k-Spanner for $k \geq 3$. We assume throughout that all edges have length 1; while much previous work has dealt with spanners with arbitrary edge lengths, our results (and all previous results on optimizing the degree) are specific to uniform edge lengths. Handling general edge lengths is an intriguing open problem.

As we also note later, it is easy to see that any k-spanner must have maximum degree at least $\Delta^{1/k}$ (simply to span the edges incident to the node of maximum degree). Thus, simply outputting the original graph is a $\Delta^{1-1/k}$ approximation. We beat the trivial algorithm, and give the following algorithmic result[3]:

▶ **Theorem 1.** *For any integer $k \geq 1$, there is an $\tilde{O}(\Delta^{(1-\frac{1}{k})^2})$-approximation for* Lowest Degree k-Spanner.

While this may seem like a rather small improvement over the trivial $\Delta^{1-1/k}$-approximation, it still requires significant technical work (possibly explaining why no nontrivial bounds were known previously). Note that in the special case of $k = 2$ our bound recovers the bound of [21], although not the improved one of [10]. This is not a coincidence: our algorithm is a modification of [21], albeit with a very different and significantly more involved analysis. We use a natural flow-based linear program in which the decision variable for each edge is interpreted as a capacity, while the spanning requirement is interpreted as requiring that for every original edge $\{u, v\}$ there is enough capacity to send 1 unit of flow along paths of length at most k (this is essentially the same LP used for directed spanners by [12, 4] but with a degree objective, and reduces to the LP used by [21] when $k = 2$).

[2] With appropriate changes to the LP, our algorithm also works for the variant in which we measure the out-degree.

[3] Our algorithm and analysis work for both the undirected and directed case with no change. The parameter k is taken to be a constant, and the $\tilde{O}$ notation hides polylogarithmic factors of the form $O(\log n(\log \Delta)^c)$ for some $c = c(k)$.

The LP rounding in [21] was a simple independent randomized rounding which ensured that every path of length 2 is contained in the spanner with probability that is at least the LP flow along that path. Since paths of length 2 with common endpoints are naturally edge-disjoint, these events are independent (for a fixed edge (u, v)), and a simple calculation shows that at least one $u - v$ path survives the rounding with probability at least $1 - 1/e$.

When $k \geq 3$ the structure of these paths becomes significantly more complicated. While we still guarantee that each flow path will be contained in the spanner with probability proportional to the amount of flow in the path, we can no longer guarantee independence, as the flow paths are not disjoint, and may intersect and overlap in highly non-trivial ways. Our main technical contribution (in the upper bound) shows that the rounding exhibits a certain dichotomy: either we can carefully prune the paths (while retaining $1/\text{polylog}(\Delta)$ flow) until they are disjoint, or the number of flow-paths that survive the rounding is concentrated around an expectation which is $\omega(1)$. This ensures that (after boosting by repeating the rounding a polylogarithmic number of rounds), every edge is spanned with high probability.

On the lower bound side, our main result is the following:

▶ **Theorem 2.** *For any integer $k \geq 3$, there is no polynomial time algorithm that can approximate* Lowest Degree k-Spanner *better than $\Delta^{\Omega(1/k)}$ unless* NP $\subseteq$ BPTIME$(2^{\text{polylog}(n)})$.

We can actually get a stronger hardness result if we assume that the input graph is directed:

▶ **Theorem 3.** *There is some constant $\gamma > 0$ such that for any integer $k \geq 3$ there is no polynomial time algorithm that can approximate* Lowest Degree k-Spanner *on directed graphs better than Δ^{γ}, unless* NP $\subseteq$ BPTIME$(2^{\text{polylog}(n)})$.

It is important to note that these hardness results do not hold if we replace Δ by n, as the algorithmic results do. The instances generated by the hardness reduction have a maximum degree that is subpolynomial in n, so the best hardness that we would be able to prove (in terms of n) would be subpolynomial (although still superpolylogarithmic). On the other hand, by phrasing the hardness in terms of Δ we not only allow direct comparisons to the upper bounds, but also allow us to use techniques (namely reductions from Label Cover and Min-Rep) that typically give only subpolynomial hardness results. Our hardness results require a mix of previous techniques and ideas, but with some interesting twists.

There is a well-developed framework (mostly put forward by Kortsarz [19] and Elkin and Peleg [15]) for proving hardness for spanner problems by reducing from Min-Rep, a minimization problem related to Label Cover that has proven useful for proving hardness (see Section 3 for the formal definition). Our reductions have two key modifications. First, we boost the degree by including many copies of both the starting Min-Rep instance and the added gadget nodes. This was unnecessary for previous spanner problems because boosting the degree was not necessary – it was sufficient to boost the number of edges by including many copies of just the gadget nodes.

The second modification is particular to the undirected case. Undirected spanner problems are difficult to prove hard because if we try to simply apply the generic framework for reducing from Min-Rep, there can be extra "fake" paths that allow the spanner to bypass the Min-Rep instance altogether. Elkin and Peleg [16] showed that for basic (min-cardinality rather than min-degree) undirected k-spanner it was sufficient to use Min-Rep instances with large girth: applying the framework to those instances would yield hardness for basic k-spanner. But they left open the problem of actually proving that Min-Rep with large girth was hard. This was proved recently [11] by subsampling the Min-Rep instance to get rid of short cycles while still preserving hardness., finally proving hardness for basic k-spanner.

We might hope LDkS is similar enough to basic k-spanner that we could just apply the generic reduction to Min-Rep with large girth. Unfortunately this does not work, since the steps we take to boost the degree end up introducing short cycles even if the starting Min-Rep instance has large girth (unlike the reduction used for basic k-spanner [16]). So we might instead hope that we could simply use the *ideas* of [11], and subsample after doing the reduction rather than before. Unfortunately this does not work either. Instead, we must do both: apply the normal reduction to the special (already subsampled) Min-Rep instances from [11], and then do an extra, separate round of subsampling on the reduction. In other words, we must sample both the Min-Rep instance itself *and* the graph obtained by applying the generic reduction to these already sampled instances.

1.2 Related Work

There has been a huge amount of work on graph spanners, from their original introduction in the late 80's [22, 23] to today. The best bounds on the tradeoff between stretch and space were reached by Althöfer et al. [1].

Most of the work since then has been on extending these tradeoffs (e.g. including additive stretch [2, 9], fault-tolerance [8, 13], or average stretch [6]) or considering algorithmic aspects such as allowing fast distance queries [26] or extremely fast constructions [18].

In parallel with this, there has been a line of work on approximating graph spanners. This was initiated by Kortsarz and Peleg, who gave an $O(\log(|E|/|V|))$-approximation for the sparsest 2-spanner problem [20] and then an $O(\Delta^{1/4})$-approximation for Lowest Degree 2-Spanner [21]. This was followed by upper bounds by Elkin and Peleg [17] for a variety of related spanner problems including LD2S (although not LDkS).

With the exception of [21], one feature that the approximation algorithms for spanners have shared with the global bounds on spanners has been the use of purely combinatorial techniques. Kortsarz and Peleg introduced the use of linear programming for spanners [21], but this was a somewhat isolated example. More recently, linear programming relaxations have become a dominant technique, and have been used for transitive closure spanners [5], directed spanners [12, 4], fault-tolerant spanners [12, 13], and LD2S [10]. In this paper we use a rounding scheme similar to [21] (with a much more complicated analysis) and an LP that is a degree-based variant of the flow-based LP introduced by [12] (an earlier use of flow-based LPs for approximating spanners is [14]).

On the hardness side, the first results were due to Kortsarz [19] who proved $\Omega(\log n)$-hardness for the basic k-spanner problem (for constant k) and $2^{\log^{1-\epsilon} n}$-hardness for a weighted version. These results were pushed further by Elkin and Peleg [15], who proved the same $2^{\log^{1-\epsilon} n}$-hardness for a collection of spanner problems including directed k-spanner. Separately, Kortsarz and Peleg proved logarithmic hardness for LD2S [21]. Proving strong hardness for basic k-spanner remained open until recently, when Dinitz, Kortsarz, and Raz proved it by showing that Min-Rep is hard even when the instances have large girth [11]. They accomplished this through careful subsampling, which we push further by subsampling both before and after the reduction.

1.3 Preliminaries

We now give some basic formal definitions which will be useful throughout this paper. Given an unweighted graph $G = (V, E)$, we let $d_G(u, v)$ denote the shortest-path distance from u to v in G, i.e. the minimum number of edges in any path from u to v (note that if G is directed this may be asymmetric). The *girth* of a graph is the minimum number of edges in any cycle

in the graph. We use the notation $e \sim v$ to indicate that e is incident on v, and the notation $p : u \rightsquigarrow v$ to indicate that p is a path from u to v. We think of paths as tuples of edges, and denote by $(p)_i$ the ith edge in a path p. For integer k, we will use $[k]$ to denote the set $\{1, 2, \ldots, k\}$.

A *k-spanner* of G is a subgraph H of G in which $d_H(u, v) \leq k \cdot d_G(u, v)$ for all $u, v \in V$. The value k is referred to as the *stretch* of the spanner. The fundamental problem that we are concerned with is the following:

▶ **Definition 4.** Suppose we are given an unweighted graph G and a stretch parameter k. The problem of computing the k-spanner that minimizes the maximum degree is Lowest Degree k-Spanner.

2 The algorithm

We now present our approximation algorithm for LDkS, proving Theorem 1. It is not hard to see that a subgraph with maximum degree D can only be a k-spanner if the original graph has degree at most $\sum_{i=1}^{k} D^i = (1 + o(1))D^k$ (the maximum number of possible paths of length $\leq k$ starting from a given node in the spanner). Therefore, we have

▶ Observation 5. In a graph with maximum degree Δ, any k-spanner must have maximum degree at least $\Omega(\Delta^{1/k})$.

2.1 LP relaxation, rounding, and approximation guarantee

Our algorithm uses the following natural LP relaxation:

$$\begin{aligned}
\min \quad & d & & \\
\text{s.t.} \quad & \textstyle\sum_{e \sim v} x_e \leq d & \forall v \in V & \quad (1)\\
& \textstyle\sum_{p : u \rightsquigarrow v, |p| \leq k} y_p = 1 & \forall (u, v) \in E & \quad (2)\\
& x_e \geq \sum_{\substack{p : u \rightsquigarrow v, |p| \leq k \\ p \ni e}} y_p & \forall (u, v), e \in E & \quad (3)\\
& x_e, y_p \geq 0 & \forall e, p & \quad (4)
\end{aligned}$$

Note that this LP has polynomial size when k is constant, and can even be solved in polynomial time when k is superconstant [12].

Recall that in a k-spanner, it is sufficient to span every edge by a path of length k. Note that for any edge $(u, v) \in E$ there may be multiple paths in the spanner spanning this edge. However, we can always pick one such path per edge. In the intended (integral) solution to the above formulation, x_e is an indicator for whether e appears in the spanner, and y_p is an indicator for the unique spanner path we assign to (u, v) (it could even be just the edge itself, if $p = (u, v)$). Thus, combined with the above observation, we have

▶ Observation 6. In a graph with maximum degree Δ, in which the optimal solution to the above LP is d_{LP}, any k-spanner (including the optimum spanner) must have maximum degree at least $\Omega(\max\{d_{\text{LP}}, \Delta^{1/k}\})$.

We apply a rather naïve rounding algorithm to the LP solution, which can be thought of as a natural extension of the rounding in [21] for LD2S:

- Independently add each edge $e \in E$ to the spanner with probability $x_e^{1/k}$.

The heart of our analysis is showing that in the subgraph this produces, every original edge is spanned with probability at least $\tilde{\Omega}(1)$. It is then only a matter of repeating the above algorithm a polylogarithmic number of times to ensure that every edge is spanned w.h.p. This will only incur a polylogarithmic factor in the degree guarantee. The following lemma gives an easy bound on the expected degree of any vertex in the above rounding:

▶ **Lemma 7.** *Let H be the subgraph obtained from the above rounding, and let d_{OPT} be the smallest possible degree of a k-spanner of $G = (V, E)$. Then every vertex in H has expected degree at most $O(d_{\text{OPT}}\Delta^{(1-1/k)^2})$.*

Proof. By linearity of expectation, the expected degree of any $v \in V$ is

$$\begin{aligned}
\textstyle\sum_{e\sim v} x_e^{1/k} &\leq (\deg_G(v))^{1-1/k} \left(\textstyle\sum_{e\sim v} x_e\right)^{1/k} && \text{by Jensen's inequality}^4 \\
&\leq \Delta^{1-1/k} d_{\text{LP}}^{1/k} \\
&= O\left(d_{\text{OPT}}\Delta^{1-1/k} \cdot \frac{d_{\text{LP}}^{1/k}}{\max\{d_{\text{LP}}, \Delta^{1/k}\}}\right) && \text{by Observation 6}
\end{aligned}$$

Noting that the last expression is maximized when $d_{\text{LP}} = \Delta^{1/k}$, we get

$$\textstyle\sum_{e\sim v} x_e^{1/k} = O(d_{\text{OPT}}\Delta^{1-1/k} \cdot (\Delta^{1/k})^{1/k-1}) = O(d_{\text{OPT}}\Delta^{(1-1/k)^2}).$$

◀

▶ Remark. Note that a simple Chernoff bound says that all degrees will be concentrated around their respective expectations, as long as the expectations are sufficiently large (say $\geq 3\ln n$). Since we repeat the basic algorithm at least $3\ln n$ times, the concentration argument can be applied to the total number of incident edges added, with multiplicities.

Thus, the crux of the analysis is to show that, indeed, every edge will be spanned with some reasonable probability.

2.2 Sketch of proof of correctness

Suppose, for simplicity, that for an edge $(u, v) \in E$, all the contribution in (2) (the spanning constraint) comes from paths of length exactly k. First, suppose all the paths with non-zero weight y_p in (2) are disjoint. For every edge e in such a path p we have, from (3), that $x_e \geq y_p$. Therefore, the probability that such a path survives (i.e. all the edges in it are retained in the rounding) is $\prod_{e\in p} x_e^{1/k} \geq y_p$. Denoting by $P(= P(u, v))$ the set of such paths, by disjointness these events are independent, and therefore we have

$$\text{Prob}[(u, v) \text{ is spanned}] \geq 1 - \prod_{p\in P}(1 - y_p) \geq 1 - \prod_{p\in P} e^{-y_p} = 1 - e^{\sum_{p\in P} y_p} = 1 - 1/e.$$

Thus, repeating this process $O(\log n)$ times, all such edges will be spanned w.h.p.

However, $u \rightsquigarrow v$ paths of length ≥ 3 need not be disjoint in general. We may assume that all paths $p \in P$ have some fixed length $k' \in [k]$ and are tuples of the form $(e_i)_{i\in I} \in \prod_{i\in I} E_i$ for some disjoint edge sets $E_1, \ldots, E_k \subset E$ (see Lemma 8). Consider the extreme example where $k' = k$ and the flow is distributed evenly over all possible paths of the form $u - v_1 - v_2 - \ldots - v_{k-1} - v$ for $v_i \in V_i$, where $\{V_i \mid i \in [k]\}$ is an equipartition of $V \setminus \{u, v\}$.

[4] or Hölder's inequality

Here, the amount of flow through each edge in the first and last layers is roughly $(k-1)/n$, and the amount of flow through any edge in the other layers is roughly $((k-1)/n)^2$. Thus, in the worst case, edges in the first and last layers will have values $x_e = (k-1)/n$ and in the other layers $x_e = ((k-1)/n)^2$. It is easy to see that the number of edges from u to V_1 that are still present (after the rounding) is concentrated around $(n/(k-1))^{1-1/k}$ (since each outgoing edge from u is retained independently with probability $((k-1)/n)^{1/k}$). Similarly, every vertex in layers $i = 2, \ldots, k-2$ will retain $\sim (n/(k-1))^{1-2/k}$ edges to the next layer, creating a total of $(n/(k-1))^{1-1/k+(1-2/k)(k-2)}$ paths from u to V_{k-1}, an $(n/(k-1))^{-1/k}$ fraction of which will continue to v. Thus, not only is (u,v) spanned after the rounding, it is spanned by $\sim (n/(k-1))^{(k^2-3k+2)/k}$ different paths (unlike the disjoint case, where only a constant number of paths survive).

Thus intuitively we have two scenarios: either the paths are disjoint, or they overlap, and a large number of them survive (both in expectation and w.h.p. due to concentration). However, this is not easy to formalize (moreover, we note that on an edge-by-edge basis, gradually merging two paths does not monotonically increase the probability that at least one path survives). To greatly simplify the formalization of this dichotomy, we prune the paths to achieve near-regularity in the LP values and combinatorial structure of the flow. To describe the outcome of the pruning, we need to introduce one more notation: Given a set of paths P' and (small) set of edges S, we denote by $m_{P'}(S)$ the number of paths $p \in P'$ such that p contains S. For example, $m'_P(\emptyset) = |P'|$ and for any path $p \in P'$ (considering p as a set of edges), $m_{P'}(p) = 1$.

The pruning procedure, which is only needed for the analysis, is an extension of standard pruning techniques (e.g. pruning to make a bipartite graph nearly regular), and is summarized in the following lemma, whose proof will appear in the full version of this paper.

▶ **Lemma 8.** *There exists a function f such that for any vertices $u, v \in V$ and set P of paths from u to v of length at most k such that $\sum_{p \in P} y_p \geq 1/\text{polylog}(\Delta)$, there exists a subset of paths $P' \subseteq P$ satisfying:*

- *For some $k' \in [k]$, all paths in P' have length k'.*
- *All the paths in P' are tuples in $\prod_{i=1}^{k'} E_i$ for some pairwise disjoint collection of sets $E_1, \ldots, E_{k'} \subset E$.*
- *There exists some $y_0 > 0$ such that every path has weight $y_p \in [y_0, 2y_0]$. Furthermore, $y_0|P'| \geq 1/(\log \Delta)^{f(k)}$.*
- *There exists a positive integer vector $(m_I)_{I \subseteq k'}$ such that $m_{P'}((e_i)_{i \in I}) \in [m_I, m_I(\log \Delta)^{f(k)}]$ for every index set $\emptyset \neq I \subseteq [k']$ and every I-tuple $(e_i)_{i \in I} \in \prod_{i \in I} E_i$ which is contained in some path in P'. (Note that if $e \in (e_i)$ then $m((e_i)) \leq m(e)$ and therefore $m_I \leq m_i(\log \Delta)^{f(k)}$ for $i \in I$).*

We note that if $\prod_{i=1}^{k'} m_{\{i\}} \leq \text{polylog}(\Delta)$, this is quite close to the disjoint paths case (where $m_{\{i\}} = 1$), and can be analyzed accordingly. The following Lemma gives the relevant result for this case.

▶ **Lemma 9.** *Let P' be the set of paths given by Lemma 8, and suppose $\prod_{i=1}^{k'} m_{\{i\}} < (\log \Delta)^{g(k)}$. Then with probability at least $1/(\log \Delta)^{h(k)}$ (for some function h), at least one path in P' survives the rounding.*

Proof. For the sake of the analysis, let us prune the paths even further. Go through every level E_i for $i = 1, \ldots, k'$ sequentially, and for every $e \in E_i$, choose exactly one (undeleted) path that contains e and delete all other paths containing e. Since for all $e \in E_i$

we have $m_{P'}(e) \leq m_{\{i\}}(\log \Delta)^{f(k)}$, in each level we retain at least a $1/(m_{\{i\}}(\log \Delta)^{f(k)})$-fraction of paths. Therefore, we end up with a new collection of paths $P^* \subseteq P'$ such that $|P^*| \geq |P'|/(\log \Delta)^{g(k)+k'f(k)}$, and the paths in P^* are edge-disjoint.

The analysis is now straightforward. Every path $p \in P^*$ is retained with probability

$$\prod_{e \in p} x_e^{1/k} \geq \prod_{e \in p} y_p^{1/k} \geq y_0^{k'/k} \geq (|P'|(\log \Delta)^{f(k)})^{-k'/k}$$

There are $|P^*|$ such paths, and each survives independently of the rest, therefore, at least one path in P^* survives with probability

$$\begin{aligned}
1 - (1 - \prod_{i=1}^{k'} x_i^{1/k})^{|P^*|} &\geq 1 - \exp(-|P^*| \prod_{i=1}^{k'} x_i^{1/k}) \\
&\geq 1 - \exp(-|P'|^{(k-k')/k} (\log \Delta)^{-(f(k)k'/k + g(k) + k'f(k))}) \\
&\geq 1 - \exp(-(\log \Delta)^{-(1+1/k)(f(k)+g(k))}) \\
&= (1 - o(1))(\log \Delta)^{-(1+1/k)(f(k)+g(k))}.
\end{aligned}$$

◀

We can also easily deal with the case $m_{\{i\}} \geq |P'|/\text{polylog}(\Delta)$, which indicates that in some layer i, the paths are concentrated in a small number of edges, by choosing just one edge $e \in E_i$, contracting this edge, and deleting all paths that do not use e (see the proof of Theorem 11). Thus, the main case we have to deal with is the intermediate case, where there is non-negligible overlap ($\prod m_{\{i\}}$ is not too small), but also no edges have too large a load (no $m_{\{i\}}$ is too close to $|P'|$). It is not hard to show that in this case the expected number of paths will be large, but showing concentration is more challenging. This constitutes the bulk of the technical analysis.

To briefly describe this part of the analysis, consider a single edge $e \in E_i$. We know this edge is contained in $m(\{e\})$ paths in P', and each of these paths has weight $y_p \in [y_0, 2y_0]$. Therefore, by constraint (3) and Lemma 8, we have

$$x_e \geq m(\{e\})y_0 \geq m_{\{i\}}y_0 \geq m_{\{i\}}/(|P'|(\log \Delta)^{f(k)}). \tag{5}$$

Suppose instead of sampling each edge independently with probability $x_e^{1/k}$, we retained any edge $e \in E_i$ with probability $x_i^{1/k}$ for

$$x_i := m_{\{i\}}/(|P'|(\log \Delta)^{f(k)}),$$

and let Y be the number of paths in P' that survive this rounding. This is clearly a lower bound for the number of paths retained in our original rounding algorithm (we can think of the modified rounding as first applying the original rounding, and then subsampling the edges even further). Note that $\mathbb{E}[Y] = |P'|^{1-k'/k}(\prod_i m_{\{i\}})^{1/k'}/(\log \Delta)^{f(k)k'/k}$, so, as we've mentioned, if $\prod_i m_{\{i\}}$ is large, then $\mathbb{E}[Y]$ will also be large. By Chebyshev's inequality, we can bound the probability that $Y = 0$ by

$$\text{Prob}[Y = 0] \leq \text{Prob}[Y < \mathbb{E}[Y]/2] \leq \text{Prob}\left[(Y - \mathbb{E}[Y])^2 > \tfrac{1}{4}(\mathbb{E}[Y])^2\right] < \frac{4\text{Var}[Y]}{(\mathbb{E}[Y])^2}$$

Thus, to prove, say, that $\text{Prob}[Y = 0] < \frac{1}{2}$, it suffices to show that

$$(\text{Var}[Y] =)\ \mathbb{E}[Y^2] - (\mathbb{E}[Y])^2 < \frac{1}{8}(\mathbb{E}[Y])^2. \tag{6}$$

While the proof of this bound is somewhat technical, it is greatly simplified by the pruning phase, which allows us to bound the variance directly as a function of the m_I values without having to analyze the combinatorial structure of the flow. The result for the main case is given by the following lemma, whose proof is deferred to the full version due to space constraints:

▶ **Lemma 10.** *Let P' be the set of paths given by Lemma 8. Then if $\prod_{i=1}^{k'} m_{\{i\}} \geq (\log \Delta)^{g(k)}$, and for every $i \in [k']$ we have $m_{\{i\}} \leq |P'|(\log \Delta)^{-g(k)}$, where $g(k) \geq (4k+2)f(k)$ then (6) holds.*

Finally, we combine these three components to give our correctness guarantee:

▶ **Theorem 11.** *Let P' be a collection of $u \rightsquigarrow v$ paths as in Lemma 8, then with probability at least $1/(\log \Delta)^{l(k)}$ (for some function l), at least one path in P' survives the rounding.*

Proof. First, consider the case where $m_{\{i\}} \leq |P'|(\log \Delta)^{-(4k+2)f(k)}$ for every $i \in [k']$. In this case, if $\prod_{i=1}^{k'} m_{\{i\}} \geq (\log \Delta)^{(4k+2)f(k)}$, then the theorem follows directly from our second moment argument and Lemma 10. If $\prod_{i=1}^{k'} m_{\{i\}} < (\log \Delta)^{(4k+2)f(k)}$, on the other hand, then the theorem follows from Lemma 9.

On the other hand, if there does exist some $i \in [k']$ such that $m_{\{i\}} \geq |P'|(\log \Delta)^{-(4k+2)f(k)}$, then the above analysis breaks down. In this case, choose any edge $e \in E_i$, and note that (by (5))

$$x_e \geq m_{\{i\}}/(|P'|(\log \Delta)^{f(k)}) \geq (\log \Delta)^{-(4k+3)f(k)}.$$

Suppose $e = (s,t)$. In the undirected case, we have a minor technical detail: we choose the direction (s,t) or (t,s) which contains at least $x'_e/2$ flow in P, say (s,t). Let P'_e be the set of paths in P' that use (s,t) (in this direction) (in the directed case, P'_e is just the set of paths in P' that use e). Then every path in P'_e consists of three parts: a $u \rightsquigarrow s$ prefix of length $i-1$, the edge (s,t), and a $t \rightsquigarrow v$ suffix of length $k'-i$. Let P''_e be the set of contracted paths $\{p/\{e\} \mid p \in P'_e\}$ in the contracted graph $G/\{e\}$. The paths in P''_e are clearly in a one-to-one correspondence with the paths in P'_e. Note that the paths P''_e satisfy all the properties given by Lemma 8 with $(4k+4)f(k)$ in place of $f(k)$ (where we define $m_{P''_e}(S) := m_{P'}(S \cup \{e\})$).

The original rounding will retain edge e with probability at least $(\log \Delta)^{-(4+3/k)f(k)}$. However, by induction on k, there is also a $(\log \Delta)^{-l(k-1)}$ probability that some (contracted) $u \rightsquigarrow v$ path in P''_e will survive. Since this event is independent of the event where e is retained, we have that at least one path in P'_e will survive with probability at least $(\log \Delta)^{-(4+3/k)f(k)-l(k-1)}$. ◀

Applying the above theorem to the set P' of paths given by Lemma 8 applied to the set P of all $u \rightsquigarrow v$ paths of length at most k, it follows that (one iteration of) our rounding algorithm spans every edge with reasonably large probability:

▶ **Corollary 12.** *Given a solution to the LP relaxation, our rounding algorithm spans every edge (by a path of length at most k) with probability at least $1/\text{polylog}(\Delta)$.*

3 Hardness of Approximation

Our reductions are based on the framework developed by [19, 15]. Our hardness bounds rely on the Min-Rep problem. In Min-Rep we are given a bipartite graph $G = (A, B, E)$ where A is partitioned into groups $A_1, A_2, \ldots, A_r$ and B is partitioned into groups $B_1, B_2, \ldots, B_r$, with the additional property that every set A_i and every set B_j has the same size (which we

will call $|\Sigma|$ due to its connection to the alphabet of a 1-round 2-prover proof system). This graph and partition induces a new bipartite graph G' called the *supergraph* in which there is a vertex a_i for each group A_i and similarly a vertex b_j for each group B_j. There is an edge between a_i and b_j in G' if there is an edge in G between some node in A_i and some node in B_j. A node in G' is called a supernode, and similarly an edge in G' is called a superedge.

A REP-cover is a set $C \subseteq A \cup B$ with the property that for all superedges $\{a_i, b_j\}$ there are nodes $a \in A_i \cap C$ and $b \in B_j \cap C$ where $\{a, b\} \in E$. We say that $\{a, b\}$ *covers* the superedge $\{a_i, b_j\}$. The goal is to construct a REP-cover of minimum size.

We say that an instance of Min-Rep is a YES instance if $OPT = 2r$ (i.e. a single node is chosen from each group) and is a NO instance if $OPT \geq 2^{\log^{1-\epsilon} n} r$. We will sometimes refer to the hardness gap (in this case $2^{\log^{1-\epsilon} n}$) as the *soundness* s, due to the connection between Min-Rep and proof systems.

▶ **Theorem 13** ([19]). *Unless* NP $\subseteq$ DTIME$(2^{\text{polylog}(n)})$*, for any constant* $\epsilon > 0$ *there is no polynomial-time algorithm that can distinguish between YES and NO instances of Min-Rep. This is true even when the graph and the supergraph are regular, and both the supergraph degree and* $|\Sigma|$ *are polynomial in the soundness.*

In the basic reduction framework we start with a Min-Rep instance, and then for every group we add a vertex (corresponding to the supernode) which is connected to vertices in the group using paths of length approximately $k/2$. We then add an edge between any two supernodes that have a superedge in the supergraph. So there is an "outer" graph corresponding to the supergraph, as well as an "inner" graph which is just the Min-Rep graph itself. The basic idea is that the only way to span a superedge is to use a path of length k that goes through the Min-Rep instance, in which case the Min-Rep edge that is in this path corresponds to nodes in a valid REP-cover. So if we are in a YES instance there is a small REP-cover and thus a small spanner, while if we are in a NO instance every REP-cover is large and thus the spanner must have many edges in order to span the superedges.

In [15] and [19] this framework is used to prove hardness of approximation when the objective is the number of edges by creating many copies of the outside nodes (i.e. the supergraph), all of which are connected to the same inner nodes (Min-Rep graph). This forces the number of edges used in the spanner to essentially equal the size of a valid REP-cover, as all other edges used by the spanner become lower order terms. We reverse this, by creating many copies of the inner Min-Rep graph. If we simply connect a single copy of the outer graph we run into a problem, though: each superedge can be spanned by paths through *any* of the copies. There is nothing that forces it to be spanned through *all* of them, and thus nothing that forces degrees to be large. We show how to get around this by creating many copies of both the inner and the outer graph, but using asymptotically more copies of the inner graph than the outer.

3.1 Directed LDkS

We now consider the directed setting, but due to space constraints only give an outline.

Suppose we are given a bipartite Min-Rep instance $\widetilde{G} = (A, B, \widetilde{E})$ with associated supergraph $G' = (U, V, E')$. For any vertex $w \in U \cup V$ we let $\Gamma(w)$ denote its group. So $\Gamma(u) \subseteq A$ for $u \in U$, and $\Gamma(v) \subseteq B$ for $v \in V$. We will assume without loss of generality that G' is regular with degree $d_{G'}$ and $\widetilde{G}$ is regular with degree $d_{\widetilde{G}}$. Our reduction will also use a special bipartite regular graph $H = (X, Y, E_H)$, which will simply be the directed complete bipartite graph with $|X| = |Y|$. Let d_H denote the degree of a node in H, so $d_H = |X| = |Y|$. We will set all of these values to $d_{G'} + 2|\Sigma| + 1$.

Our LDkS instance $G = (V_G, E_G)$ will be a combination of these three graphs. Let $k_L = \lfloor \frac{k-1}{2} \rfloor$, and let $k_R = \lceil \frac{k-1}{2} \rceil$. The four sets of vertices are

$$V^L_{out} = U \times X \times [k_L] \qquad V^R_{out} = V \times Y \times [k_R]$$
$$V^L_{in} = A \times E_H \qquad V^R_{in} = B \times E_H.$$

The actual vertex set V_G of our LDkS instance G will be $V^L_{out} \cup V^R_{out} \cup V^L_{in} \cup V^R_{in}$. We say that an outer node is *maximal* if its final coordinate is maximal (k_L for nodes in V^L_{out} or k_R for nodes in V^R_{out}), and we say that an outer node is *minimal* if its final coordinate is 1.

Defining the edge set is a little more complex, as there are a few different types of edges. We first create outer edges, which are incident on maximal outer nodes:

$$E_{out} = \{((u, x, k_L), (v, y, k_R)) : u \in U \wedge v \in V \wedge x \in X \wedge y \in Y \wedge \{u, v\} \in E' \wedge (x, y) \in E_H\}.$$

Note that if we fix x and y the corresponding outer edges form a copy of the supergraph G'. Thus these edges essentially form $|E_H|$ copies of the supergraph.

We also have inner edges, which correspond to $|E_H|$ copies of the Min-Rep instance (note that unlike the supergraph copies, these copies are vertex disjoint):

$$E_{in} = \{((a, e), (b, e)) : a \in A \wedge b \in B \wedge e \in E_H \wedge \{a, b\} \in \widetilde{E}\}.$$

We will now add edges that connect some of the outer nodes to some of the inner nodes: let

$$E^L_{con} = \{((u, x, 1), (a, (x, y))) : u \in U \wedge a \in \Gamma(u) \wedge x \in X \wedge (x, y) \in E_H\}, \text{ and}$$
$$E^R_{con} = \{((b, (x, y)), (v, y, 1)) : v \in V \wedge b \in \Gamma(v) \wedge y \in Y \wedge (x, y) \in E_H\}.$$

In other words, the minimal outer node for each (u, x) (resp. (v, y)) is connected to the inner nodes in its group in each copy of $\widetilde{G}$ that corresponds to an E_H edge that involves x (resp. y).

We now need to connect the minimal outer nodes and the maximal outer nodes. We do this by creating paths: let

$$E^L_{path} = \{((u, x, i), (u, x, i-1)) : u \in U, x \in X, i \in \{2, \dots k_L\}\}, \text{ and}$$
$$E^R_{path} = \{((v, y, i), (v, y, i+1)) : v \in V, y \in Y, i \in [k_R - 1]\}.$$

Finally, for technical reasons we need to add edges internally in each group in each copy of $\tilde{G}$: let $E^L_{group} = \{((a, e), (a', e)) : e \in E_H \wedge a, a' \in \Gamma(u) \text{ for some } u \in U\}$, and let $E^R_{group} = \{((b, e), (b', e)) : e \in E_H \wedge b, b' \in \Gamma(v) \text{ for some } v \in V\}$.

Our final edge set is the union of all of these, namely $E_{out} \cup E_{in} \cup E^L_{con} \cup E^R_{con} \cup E^L_{path} \cup E^R_{path} \cup E^L_{group} \cup E^R_{group}$.

3.1.1 Analysis

The first step is to show that if there is a small REP-cover for the original Min-Rep instance, then there is a k-spanner with low maximum degree. To do this we will use the notion of a *canonical path* for an outer edge. Consider an outer edge $((u, x, k_L), (v, y, k_R))$. A path from (u, x, k_L) to (v, y, k_R) is *canonical* it if includes $k_L - 1$ path edges, followed by a connection edge, an inner edge, another connection edge, and then $k_R - 1$ path edges. Note that any such path has length $k_L + k_R + 1 = k$, so can be used to span the outer edge. Furthermore, note that any such path corresponds to selecting two nodes (the inner nodes hit by the path) that cover the $\{u, v\}$ superedge in the original Min-Rep instance.

It is not hard to see that the *only* way to span an outer edge is either through a canonical path (which corresponds to a way of covering the associated superedge in the Min-Rep instance) or including the edge itself. This means that we can span all outer edges by using canonical paths corresponding to a REP-cover, and that this is the only way spanning outer edges. Since in a YES instance there is a REP-cover in which only a single node is selected per group, we can use those canonical paths to construct a k-spanner with maximum degree at most d_H.

▶ **Lemma 14.** *If we start with a YES instance of Min-Rep, then there is a k-spanner of G which has maximum degree at most $d_H + 1$.*

Proof. Since we are in a YES instance, for each $u \in U$ there is some $f(u) \in \Gamma(u)$ and for each $v \in V$ there is some $f(v) \in \Gamma(v)$ so that $\{f(u), f(v)\} \in \tilde{E}$ for all $\{u, v\} \in E'$. Our spanner contains all edges in E^L_{group} and E^R_{group} as well as all edges in E^L_{path} and E^R_{path}. It also contains the connection edges suggested by the REP-cover: for every $u \in U$ and $x \in X$ and $(x, y) \in E_H$, it contains the connection edge $((u, x, 1), (f(u), (x, y)))$. Similarly, for every $v \in V$ and $y \in Y$ and $(x, y) \in E_H$, it contains the connection edge $((f(v), (x, y)), (v, y, 1))$. Finally, it contains the appropriate inner edges: for every $\{u, v\} \in E'$ with $u \in U$ and $v \in V$ and every $e \in E_H$, we add the inner edge $((f(u), e), (f(v), e))$.

In this spanner, the degree of outer nodes which are not minimal is at most 2 (the 2 incident path edges), and the degree of inner nodes is at most $d_{G'} + 2|\Sigma| + 1$ (since they are incident on one connection edge, $2|\Sigma|$ group edges, and $d_{G'}$ inner edges). The degree of a minimal outer node is at most $d_H + 1$, since it is incident on 1 path edge and for each edge incident on the second coordinate in E_H it is incident to a single inner node. Thus the maximum degree of the spanner is at most $\max\{d_{G'} + 2|\Sigma| + 1, d_H + 1\} = d_H + 1$ as claimed.

It remains to show that this is indeed a valid spanner. The only edges not included are the outer edges and some of the connection edges and inner edges, so we simply need to prove that they are spanned by paths of length at most k. For connection edges this is trivial. Consider some edge $((u, x, 1), (a, (x, y))) \in E^L_{con}$. Clearly there is a path of length two that spans it: an included connection edge from $(u, x, 1)$ to $(f(u), (x, y))$, followed by a group edge from $(f(u), (x, y))$ to $(a, (x, y))$. A similar path exists (in the opposite direction) for connection edges in E^R_{con}.

Similarly, consider an inner edge $((a, e), (b, e))$ which is not in the spanner. Let $u \in U$ and $v \in V$ so that $a \in \Gamma(u)$ and $b \in \Gamma(v)$. Then $\{u, v\} \in E'$, so our spanner contains an inner edge $((f(u), e), (f(v), e))$. So there is a path of length three in our spanner from (a, e) to (b, e), namely $(a, e) \to (f(u), e) \to (f(v), e) \to (b, e)$, where the first and last edges are group edges and the middle edge is an inner edge.

Now consider an outer edge $((u, x, k_L), (v, y, k_R))$. We can span it by using a canonical path, where the first connection edge will be from $(u, x, 1)$ to $(f(u), (x, y))$, the inner edge will be from $(f(u), (x, y))$ to $(f(v), (x, y))$, and the second connection edge will be from $(f(v), (x, y))$ to $(v, y, 1)$ (this fixes the path edges used as well). Note that all of these edges exist in the spanner, since the connection edges are included by construction and the inner edge must exist because this is a YES instance, i.e. because $\{f(u), f(v)\} \in \tilde{E}$ for all $\{u, v\} \in E'$. Thus this is indeed a path in the spanner, and it clearly has length k. ◀

On the other hand, since in a NO-instance there are no small REP-covers, any spanner must include either many canonical paths or many outer edges. This lets us prove that in this case every k-spanner has some node with large degree.

▶ **Lemma 15.** *If we start with a NO instance on Min-Rep, then every k-spanner of G has maximum degree at least $(s/3)d_H$*

Proof. We will prove the contrapositive, that if there is a k-spanner of G with maximum degree less than $(s/3)d_H$ then there is a REP-cover of size less than $s(|U| + |V|)$ (and thus we did not start with a NO instance). Let $\hat{G}$ be such a spanner. We create a bucket $B_{(x,y)}$ for each edge $(x, y) \in E_H$, which will contain a collection of outer edges and connection edges that are in $\hat{G}$. For each outer edge $((u, x, k_L), (v, y, k_R))$ that is in $E(\hat{G})$, we add it to the bucket $B_{(x,y)}$. Similarly, for each connection edge $((u, x, 1), (a, (x, y)))$ that is in $E^L_{con} \cap E(\hat{G})$ we add it to $B_{(x,y)}$, as well as each connection edge $((b, (x, y)), (v, y, 1)) \in E^R_{con} \cap E(\hat{G})$. Since $\hat{G}$ has maximum degree less than $(s/3)d_H$, the total number of edges in buckets (i.e. the total number of outer and connection edges in $\hat{G}$) is less than $|U||X|(s/3)d_H$ (the number of outer edges) plus $|U||X|(s/3)d_H + |V||Y|(s/3)d_H$ (the number of connection edges), for a total of $|U||X|sd_H$ edges (since both G' and H are balanced and regular).

Since H is regular we know that $|X|d_H = |E_H|$. Thus there must exist some bucket with less than $s|U| = s|V|$ edges. Let $B_{(x,y)}$ be this bucket. We will create a REP-cover as follows. For each edge $((u, x, 1), (a, (x, y))) \in E^L_{con} \cap B_{(x,y)}$ we will include a and for each edge $((b, (x, y)), (v, y, 1)) \in E^R_{con} \cap B_{(x,y)}$ we will include b. For each outer edge $((u, x, k_L), (v, y, k_R))$ we will include an arbitrary vertex in $\Gamma(u)$ and an arbitrary vertex in $\Gamma(v)$ that are adjacent in $\widetilde{G}$ (such vertices must exist in order for the Min-Rep instance to be satisfiable at all). Clearly this cover has size less than $2|B_{(x,y)}| \leq 2s|U| = s(|U| + |V|)$.

It only remains to show that this is a valid cover. To see this, consider an arbitrary superedge, say $\{u, v\}$, and the associated outer edge from (u, x, k_L) to (v, y, k_R) (where here x and y are the same as in our special bucket). It is clear that by construction the only paths of length at most k which can span an outer edge are either the outer edge itself or the canonical paths. In the former case we explicitly added an arbitrary pair of nodes that cover $\{u, v\}$. In the second case, the existence of a canonical path in the spanner means that the connection edges it uses are in the bucket. This in turn means that the inner nodes they are incident on were added to the REP-cover, and since the canonical path uses the inner edge between them they must in fact cover the $\{u, v\}$ superedge. Thus we have a valid REP-cover of size less than $s(|U| + |V|)$. ◀

We can now use Lemmas 14 and 15 to prove the desired hardness for Directed LDkS.

▶ **Theorem 16.** *Unless* NP $\subseteq$ DTIME($2^{\text{polylog}(n)}$)*, there is a constant* $\gamma > 0$ *so that no polynomial time algorithm can approximate Directed LDkS to a factor better than* Δ^γ *(for any integer* $k \geq 3$*).*

Proof. Lemmas 14 and 15, when combined with Theorem 13, imply hardness of $\Omega(s)$. With the chosen value of d_H, it is easy to verify that Δ is achieved at either maximal or minimal outer nodes. The degree in G of the former is at most $d_{G'}d_H + 1 = O(d^2_{G'} + d_{G'}|\Sigma|)$, while the latter have degree at most $|\Sigma|d_H + 1 = O(d_{G'}|\Sigma| + |\Sigma|^2)$. If $k = 3$ or 4 then the the maximal nodes might also be minimal, and so have degree equal to the sum of those bounds. But for any $k \geq 3$ we have that $\Delta \leq O(d^2_{G'} + |\Sigma|^2)$. Since we specifically chose to use hard Min-Rep instances where $d_{G'}$ and $|\Sigma|$ are polynomial in s, this proves the theorem. ◀

3.2 Undirected LDkS

We now want to handle the undirected case (again, we only give an outline). This is complicated primarily because switching edges to being undirected creates new paths that the spanner might use. In the directed setting, if an outer edge was not in the spanner then the only way for it to be spanned was to use a canonical path, which essentially determined the "suggested" REP-cover for the Min-Rep instance. Once we move to the undirected setting

there is another possibility: an outer edge could be spanned by a path consisting entirely of outer edges. This was not possible with directed edges because all outer edges were directed into V^R_{out}. These new paths are problematic, since if an outer edge is spanned in this way there is no suggested REP-cover. Thus we will try to make sure that no such paths actually exist.

We will need to start with hard Min-Rep instances with some extra properties: namely, we want large supergirth and $d_{G'} \geq |\Sigma|$. This can be achieved using a simple modification of [11], giving the following lemma.

▶ **Lemma 17.** *Unless* NP $\subseteq$ BPTIME($n^{\text{polylog}(n)}$), *there is no polynomial time algorithm that can distinguish between instances of Min-Rep in which there is a REP-cover of size* $|U| + |V|$ *(i.e. a YES instance) and instances in which every REP-cover has size at least* $s(|U| + |V|)$, *even when all instances are guaranteed to have the following properties:*

1. *The girth of the supergraph is larger than* $k + 1$,
2. *There is some value* $d_{G'}$ *so that all degrees in the supergraph are within a factor of* 2 *of* $d_{G'}$,
3. $s, d_{G'}$, *and* $|\Sigma|$ *are all polynomials of each other, and*
4. $d_{G'} \geq |\Sigma|$.

We will also use a balanced regular bipartite graph H as before, but instead of being the (directed) complete bipartite graph, H will be a balanced regular bipartite graph of girth at least $k + 2$ and degree d_H (note that such graphs exist as long as the number of nodes $n_H = |X| + |Y| = 2|X|$ is sufficiently large, e.g. as long as $n_H d_H \leq n_H^{1+\frac{1}{3k^2}}$ [3]). We will set $d_H = d_{G'}$, so the number of outer edges incident on each maximal outer node of G is $d = d_H d_{G'} = d^2_{G'}$.

We start with the same graph G as in the directed setting (although with undirected edges, and using Min-Rep instances from Lemma 17).

We will then subsample in essentially the same way as [11]: for every outer edge $\{(u, x, k_L), (v, y, k_R)\}$ we will flip an independent coin, keeping the edge with probability $p = \frac{\alpha \log |\Sigma|}{d}$ and removing it with probability $1 - p$ (we will set $\alpha = d^{\frac{k+2}{2(k+1)}}/(4 \log |\Sigma|)$). If we remove it we will also remove the associated inner edges, i.e. we will remove all inner edges of the form $\{(a, \{x, y\}), (b, \{x, y\})\}$ where $a \in \Gamma(u)$ and $b \in \Gamma(v)$. This gives us a new graph G_α.

Call an outer edge of G_α *bad* if it is part of a cycle in G_α consisting only of outer edges of length at most $k + 1$. We will see that there are not too many bad edges, so we then create our final instance of LDkS by removing all bad edges (and associated inner edges) from G_α, giving us a new graph $\widehat{G_\alpha}$. Intuitively $\widehat{G_\alpha}$ is essentially the same as G_α, since there are so few bad edges in G_α.

3.2.1 Analysis

We can still build a spanner using canonical paths corresponding to a REP-cover of each subsampled instance, so if we start with a YES instance we can still build a spanner of $\widehat{G_\alpha}$ with small maximum degree. This is essentially the same as Lemma 14.

▶ **Lemma 18.** *If we started out with a YES instance of Min-Rep, there is a* k*-spanner of* $\widehat{G_\alpha}$ *with maximum degree at most* $\max\{d_H + 1, d_{G'}^{\frac{1}{k+1}} + 2|\Sigma| + 1\}$.

For each outer edge $((u, x, k_L), (v, y, k_R))$ in G, call a path from (u, x, k_L) to (v, y, k_R) *bad* if it contains only outer edges and has length at most k (and larger than 1). So an outer

edge is bad if and only if there is a bad path between its endpoints. We begin by analyzing the number of bad paths for any fixed outer edge in the original construction G (before subsampling). The trivial bound would be d^{k-1}, but because G' and H both have large girth we can do better. This is the reason we needed to start out with already subsampled instances of Min-Rep (i.e. why we had to start with instances based on [11] rather than generic hard Min-Rep instances, like those from Theorem 13).

▶ **Lemma 19.** *For any outer edge, the number of bad paths is at most* $O(4^k d^{\frac{k-1}{2}})$.

Lemma 19 now allows us to upper bound the number of bad edges in G_α, since we set α to be low enough that we expect all of the bad paths in G to be missing at least one edge in G_α.

▶ **Lemma 20.** *With probability at least* $3/4$ *the number of outer edges in* G_α *that are bad is at most* $|U| \cdot |X| \cdot d_H$

Recall that our construction started with $|E_H| = |X|d_H$ copies of the original Min-Rep instance, and each outer edge is associated with a single such instance. So in G_α the average instance has at most $|U|$ bad edges, and thus by Markov at least $|E_H|/2$ of the instances have at most $2|U|$ bad edges. It is well-known that removing only $|U|$ superedges of a Min-Rep instance affects the size of the optimal REP-cover in a NO instance by at most a constant factor (see e.g. [11] for a proof of this). So $\widehat{G_\alpha}$ is essentially G_α. So now that there are no bad edges, the only ways to span an outer edge in $\widehat{G}_\alpha$ are the edge itself or a canonical path, so we are essentially back to the directed case (except that we can only use $1/2$ of the $|E_H|$ Min-Rep instances to prove our bound, but that is plenty). This implies that in a NO instance all spanners must have large maximum degree, through an analysis similar to Lemma 15.

▶ **Lemma 21.** *It we started out with a NO instance of Min-Rep, any* k*-spanner of* $\widehat{G_\alpha}$ *must have a node of degree at least* $\tilde{\Omega}(d_h \cdot d_{G'}^{\frac{1}{2(k+1)}})$.

The main hardness theorem is now implied by the chosen parameters.

▶ **Theorem 22.** *Unless* NP $\subseteq$ BPTIME$(n^{\text{polylog}(n)})$*, there is no algorithm that can approximate* Lowest Degree k-Spanner *on undirected graphs to a factor better than* $\Delta^{\Omega(1/k)}$ *(for any integer* $k \geq 3$*).*

References

1. Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete Comput. Geom.*, 9(1):81–100, 1993.
2. Surender Baswana, Telikepalli Kavitha, Kurt Mehlhorn, and Seth Pettie. Additive spanners and (α, β)-spanners. *ACM Trans. Algorithms*, 7(1):5:1–5:26, December 2010.
3. Mohsen Bayati, Andrea Montanari, and Amin Saberi. Generating random graphs with large girth. In *SODA'09*, pages 566–575, 2009.
4. Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Improved approximation for the directed spanner problem. In *ICALP (1)*, pages 1–12, 2011.
5. Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. In *SODA'09*, pages 932–941, 2009.
6. T.-H. Hubert Chan, Michael Dinitz, and Anupam Gupta. Spanners with slack. In *Proceedings of the 14th Annual European Symposium on Algorithms*, ESA, pages 196–207, 2006.

7 Barun Chandra, Gautam Das, Giri Narasimhan, and José Soares. New sparseness results on graph spanners. *International Journal of Computational Geometry and Applications*, 5(1):125–144, 1995.

8 S. Chechik, M. Langberg, David Peleg, and L. Roditty. Fault-tolerant spanners for general graphs. In *STOC'09*, pages 435–444, New York, NY, USA, 2009. ACM.

9 Shiri Chechik. New additive spanners. In *SODA'13*, pages 498–512, 2013.

10 Eden Chlamtáč, Michael Dinitz, and Robert Krauthgamer. Everywhere-sparse spanners via dense subgraphs. *FOCS'12*, 0:758–767, 2012.

11 Michael Dinitz, Guy Kortsarz, and Ran Raz. Label cover instances with large girth and the hardness of approximating basic k-spanner. In *ICALP'12*, pages 290–301, Berlin, Heidelberg, 2012. Springer-Verlag.

12 Michael Dinitz and Robert Krauthgamer. Directed spanners via flow-based linear programs. In *STOC'11*, pages 323–332, New York, NY, USA, 2011. ACM.

13 Michael Dinitz and Robert Krauthgamer. Fault-tolerant spanners: Better and simpler. In *PODC'11*, pages 169–178, 2011.

14 Yevgeniy Dodis and Sanjeev Khanna. Designing networks with bounded pairwise distance. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, STOC'99, pages 750–759, New York, NY, USA, 1999. ACM.

15 Michael Elkin and David Peleg. The hardness of approximating spanner problems. In *STACS*, pages 370–381, 2000.

16 Michael Elkin and David Peleg. Strong inapproximability of the basic k-spanner problem. In *ICALP*, pages 636–647, 2000.

17 Michael Elkin and David Peleg. Approximating k-spanner problems for $k > 2$. *Theor. Comput. Sci.*, 337(1-3):249–277, 2005.

18 Michael Elkin and Shay Solomon. Fast constructions of light-weight spanners for general graphs. In *In Proc. of 24th SODA*, pages 513–525, 2013.

19 Guy Kortsarz. On the hardness of approximating spanners. *Algorithmica*, 30(3):432–450, 2001.

20 Guy Kortsarz and David Peleg. Generating sparse 2-spanners. *J. Algorithms*, 17(2):222–236, 1994.

21 Guy Kortsarz and David Peleg. Generating low-degree 2-spanners. *SIAM J. Comput.*, 27(5):1438–1456, 1998.

22 David Peleg and Alejandro A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989.

23 David Peleg and Jeffrey D. Ullman. An optimal synchronizer for the hypercube. *SIAM J. Comput.*, 18(4):740–747, 1989.

24 Mohit Singh and Lap Chi Lau. Approximating minimum bounded degree spanning trees to within one of optimal. In *STOC'07*, pages 661–670, 2007.

25 Mikkel Thorup and Uri Zwick. Compact routing schemes. In *SPAA'01*, pages 1–10, New York, NY, USA, 2001. ACM.

26 Mikkel Thorup and Uri Zwick. Approximate distance oracles. *J. ACM*, 52(1):1–24, January 2005.

Improved Streaming Algorithms for Weighted Matching, via Unweighted Matching

Michael Crouch and Daniel M. Stubbs

University of Massachusetts, Amherst, U.S.

Abstract

We present a $(4+\epsilon)$ approximation algorithm for weighted graph matching which applies in the semistreaming, sliding window, and MapReduce models; this single algorithm improves the previous best algorithm in each model. The algorithm operates by reducing the maximum-weight matching problem to a polylog number of copies of the maximum-cardinality matching problem. The algorithm also extends to provide approximation guarantees for the more general problem of finding weighted independent sets in p-systems (which include intersections of p matroids and p-bounded hypergraph matching).

1998 ACM Subject Classification F.1.1 [Computation by Abstract Devices]: Models of Computation – relations between models, F.1.2 [Computation by Abstract Devices]: Modes of Computation – online computation, G.2.2 [Discrete Mathematics]: Graph Theory – graph algorithms, hypergraphs

Keywords and phrases Streaming Algorithms, Graph Matching, Weighted Graph Matching, MapReduce, Independence Systems

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.96

1 Introduction

Finding large matchings (that is, sets of edges which do not have any endpoints in common) is a pivotal problem in graph algorithms, and has seen significant recent work in a variety of "big data" models. In particular, there has been a series of papers in the semi-streaming graph model, where we have one-way read access to a stream of weighted edges, but have only $O(n \operatorname{polylog} n)$ memory (enough to store only a sparse subgraph of the input). The paper which first introduced this model [9] provided a 6-approximation algorithm for maximum weighted graph matching. This was improved to a 5.828-approximation in [15]; a 5.585-approximation in [17]; and finally the current best, a $4.911+\epsilon$-approximation in [8]. Other work has looked at maximum weighted matching in the MapReduce model [13] and the sliding-window stream model [6], and has examined more general submodular-function matching problems in the semistreaming model [2, 5].

We present an algorithm for maximum weighted matching which is applicable in all of these models and which improves on the best known approximation guarantees in all of them. Our algorithm also extends to a generalization of maximum weighted graph matching: the problem of finding maximum-weight independent sets in p-systems. p-systems are a type of independence system which generalize both matching on p-bounded hypergraphs and intersections of p matroids.

Our algorithm works by reducing a single maximum weighted matching problem to a number of unweighted matching problems, then combining the unweighted matchings according to a simple greedy heuristic. The structure of our reduction is related to an existing streaming algorithm for maximum weighted matching [8]. That algorithm partitions incoming edges into multiplicatively-spaced weight classes; it maintains a separate greedy

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 96–104

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

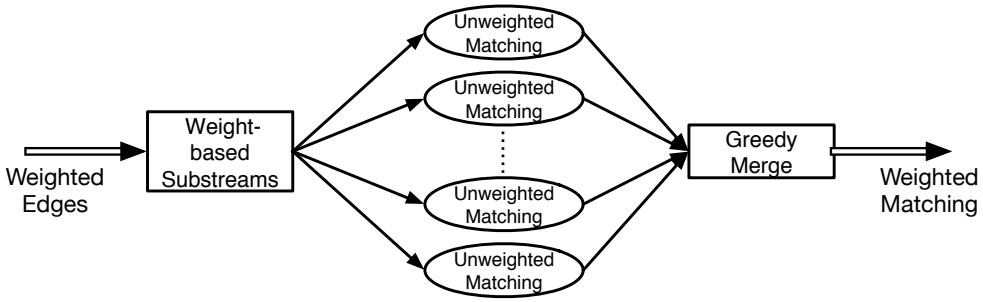


Figure 1 Block Diagram of the Weighted Matching Algorithm.

matching on the edges in each weight class. At the end of the stream, they greedily merge the matchings from largest to smallest.

Rather than partitioning the edges into disjoint weight classes, our algorithm uses weight classes that are unbounded above: classes that admit smaller-weight edges are supersets of all of the "more exclusive" classes. The stronger relationship between different weight classes created by this approach allows a more unified and global charging argument, giving an improved approximation ratio and broader applicability, including to the case of more general independence systems than just graph matchings.

In §2 we define our model more specifically and state our main results. In §3 we present the algorithm. In §4 we summarize the improvements that our that our algorithm yields for weighted matching and weighted independent set problems in the streaming, sliding window, and MapReduce models. In §6 we comment on improvements for specific cases and outline future work.

2 Definitions and Results

In this section, we define our model of "streaming reductions"; present our result for maximum weighted graph matching; recall the definitions of p-systems; and state the extensions of our results to p-systems.

2.1 Streaming Reductions

Our algorithm will operate by transforming a maximum-weight matching problem into a polylogarithmic number of maximum-cardinality matching problems (Fig. 1). This is an example of a class of reductions which we believe may be of particular interest in big data models: Turing reductions which make a polylogarithmically-bounded number of queries and which are "nonadaptive" (in the sense that the input to one query does not depend on the output of any other query).

In this model, a reduction from problem A to problem B consists of processing the input to problem A into the inputs to polylog instances of B; solving the B instances; then processing the B outputs into the output to problem A. These models form a restricted class of the approximation-preserving streaming reductions used in [4].[1]

[1] Other papers introducing reductions in the streaming model defined many-one reductions between decision problems via a generalization of string homomorphisms [3, 14]; it is not readily apparent how to apply this work to approximation problems.

These reductions are natural choices for models where resources of interest are closed under polylogarithmic blowup, including the semistreaming graph model. Requiring the subproblems to be nonadaptive allows them to be easily parallelized. The resources used by the preprocessing and postprocessing steps can be restricted to preserve the classes of interest; in our case, the preprocessing step is merely testing the weight of each edge, and the postprocessing step is a greedy merge. Many existing streaming algorithms operate by reducing to polylog nonadaptive subproblems, including the precision sampling framework [1] and Indyk/Woodruff L_p norm estimators [10].

We say that a reduction from A to B is p-approximate if, for any $\alpha \geq 1$, given an α-approximate solution to each B subproblem we can generate an αp-approximate solution to the A problem.

2.2 Main Result

Section 3 presents the proof of our main result:

▶ **Theorem 1.** *There is a $2(1+\epsilon)$-approximate nonadaptive Turing reduction from the problem of maximum-weight matching to the problem of maximum-cardinality matching. The reduction uses $O(\frac{1}{\epsilon}\log n)$ copies of maximum-cardinality matching.*

The reduction in Theorem 1 uses extremely minimal preprocessing (separating edges by weight) and minimal postprocessing (performing a greedy merge of the edge sets).

Since greedy matching provides a 2-approximation to maximum-cardinality matching, from Theorem 1 we immediately find:

▶ **Corollary 2.** *We can perform a $4(1+\epsilon)$-approximation to maximum-weight matching, using $O(\frac{1}{\epsilon}\log n)$ times the resources necessary to keep a greedy matching.*

The consequences of Corollary 2 in specific models are described in Section 4.

2.3 Independence Systems

Our algorithm extends to a class of independence systems called *p-systems*. An *independence system* is a pair (S, I) comprising a finite set S and a set I of subsets of S (the "independent sets") such that

1. $\emptyset \in I$
2. For $X \subseteq X'$, $X' \in S \Rightarrow X \in S$.

An independence system (S, I) is called a p-system if, for any $A \subseteq S$, the ratio between the largest and smallest maximal independent subsets of A is at most p. Graph matching forms a 2-system where S is the set of edges and where a set of edges is independent if no two edges share an endpoint. More generally, p-hypergraph matching is a p-system, as is the intersection of p matroids. For more detail on p-systems, see e.g. [11].

Given a p-system (S, I), the *maximum-cardinality independent set* problem is the problem of finding an independent set with the largest number of elements. Given a weight function $w : S \to \mathbb{R}_{\geq 0}$, the *maximum-weight independent set* problem is the problem of finding an independent set $X \in I$ which maximizes $\sum_{x \in X} w(x)$. These problems naturally extend the problems of finding maximum matchings on unweighted or weighted graphs.

Section 3.1 shows that Thm. 1 extends to:

▶ **Theorem 3.** *Let (S, I) be a p-system. Then there is a $p(1+\epsilon)$-approximate nonadaptive Turing reduction from the problem of maximum-weight independent set on (S, I) to the problem of maximum-cardinality independent set on (S, I). The reduction uses $O(\frac{1}{\epsilon}\log n)$ copies of maximum-cardinality independent set.*

From the definition of p-systems, a greedily maximized set is always a p-approximate maximum cardinality matching. From Theorem 3 we thus immediately find:

▶ **Corollary 4.** *We can perform a $p^2(1+\epsilon)$ approximation to maximum-weight independent set on any p-system, using $O(\frac{1}{\epsilon}\log n)$ times the resources necessary to greedily compute a maximal independent set on that p-system.*

The consequences of Corollary 4 in specific models are described in Section 4.

3 Algorithm

In this section we present a proof of Theorem 1.

Consider a graph G on vertex set V. Let $n = |V|$. Let the input E be a stream of edges from $V \times V$, where each $e \in E$ is annotated with its weight $w(e)$.

For $i \in \mathbb{Z}$, we define substreams E_i, each containing all edges with weight above threshold $(1+\epsilon)^i$:

$$E_i \triangleq \{e \in E \mid w(e) \geq (1+\epsilon)^i\} \tag{1}$$

Note that i can be negative, but we assume that the range of possible weights $w(e)$ is polynomially bounded in n, so that we only need to consider substreams for $O(\frac{1}{\epsilon}\log n)$ values of i.[2]

To perform the reduction, assume that for $\alpha > 1$ we have for each E_i some matching $C_i \subseteq E_i$ which contains at least $\frac{1}{\alpha}$ times as many elements as the maximum-cardinality matching on E_i. We then greedily construct a matching T by considering the edges in C_i in descending order of i, and at the end we output T. The top-level structure of the algorithm is summarized in Figure 1.

Consider a fixed maximum-weight matching Opt on E. For each class E_i let $T_i = T \cap E_i$ be the set of edges output from E_i.

▶ **Lemma 5.** *For each i, $|T_i| \geq \frac{1}{2\alpha}|E_i|$.*

Proof. For each class E_i let Opt_i be a maximum-cardinality matching on E_i. Our oracle returns C_i with $|C_i| \geq \frac{1}{\alpha}|\text{Opt}_i|$, and thus with $|C_i| \geq \frac{1}{\alpha}|\text{Opt} \cap E_i|$.

We greedily add as many edges as possible from C_i to T_i. Since T_i and C_i are both matchings, each edge in T_i can share endpoints with at most two edges of C_i. Thus, as long as $|T_i| < \frac{1}{2}|C_i|$, C_i contains at least one edge which is not adjacent to any edge yet in T_i.

The greedy merge can thus always add edges from C_i to T_i until $|T_i| \geq \frac{1}{2}|C_i|$. Combining with the above we then have $|T_i| \geq \frac{1}{2\alpha}|\text{Opt} \cap E_i|$. ◀

We now must argue that the cardinality constraint in Lemma 5 leads to the weight-based approximation ratio in Theorem 1.

▶ **Lemma 6.** *There exists a function f from Opt to T such that for each $e \in \text{Opt}$, $w(e) \leq (1+\epsilon)w(f(e))$ and for each $t \in T$, there are at most 2α edges $e \in \text{Opt}$ such that $f(e) = t$.*

[2] If we do not have this guarantee, we can keep track of the highest-weight edge seen so far, and discard any items with less than $2\epsilon/n$ times that weight. A matching made entirely of these discarded edges is then at most an ϵ fraction of the output weight (since the weight we output is at least the weight of the largest edge).

Proof. We define f inductively, considering $\text{OPT} \cap E_i$ in descending order by i and picking $f(e)$ from among the elements of T_i that have fewer than 2α edges already associated with them. This restriction will guarantee that $f(e)$ is from at least as high a class as e, which gives us that $w(e) \leq (1+\epsilon)w(f(e))$. By Lemma 5 there are always enough elements in T_i to avoid overcrowding.[3] ◀

Lemma 6 leads immediately to a charging argument which proves Theorem 1: every edge $e \in \text{OPT}$ is an element of the preimage $f^{-1}(t)$ for some t, and for each t

$$\sum_{e \in f^{-1}(t)} w(e) \leq \left|f^{-1}(t)\right|(1+\epsilon)w(t) \leq 2\alpha(1+\epsilon)w(t) \tag{2}$$

so we have

$$w(\text{OPT}) = \sum_{e \in \text{OPT}} w(e) = \sum_{\substack{e \in f^{-1}(t) \\ t \in T}} w(e) \leq \sum_{t \in T} 2\alpha(1+\epsilon)w(t) = 2\alpha(1+\epsilon)w(T) \tag{3}$$

3.1 Extension to p-Systems

The algorithm operates similarly for reducing maximum-weight independent sets in arbitrary p-systems to copies of the problem of finding maximum-cardinality independent sets (Theorem 3). Most of the proof is similar, with independent sets replacing matchings and with p replacing the multiplicative factor 2. The equivalent of Lemma 5 is somewhat more involved to prove:

▶ **Lemma 7.** *For each i, $|T_i| \geq \frac{1}{\alpha p}|E_i|$.*

Proof. For each class E_i let OPT_i be a maximum-cardinality independent set on E_i; we again consider an oracle which returns an independent set C_i of cardinality $|C_i| \geq \frac{1}{\alpha}|\text{OPT}_i| \geq \frac{1}{\alpha}|\text{OPT} \cap E_i|$.

E_i is a subset of a p-system and thus also forms a p-system. Now consider maximal independent sets on $C_i \cup T_i$ (recall $C_i \cup T_i \subseteq E_i$). We have that C_i is a maximal independent set of size $|C_i|$. Thus, by the definition of p-systems, no maximal independent subset of $C_i \cup T_i$ can have size less than $\frac{1}{p}|C_i|$.

The greedy merge can thus always add elements from C_i until $|T_i| \geq \frac{1}{p}|C_i|$, yielding $|T_i| \geq \frac{1}{\alpha p}|\text{OPT} \cup E_i|$. ◀

The remainder of the proof of Theorem 3 proceeds similarly to the proof of Theorem 1.

4 Extensions

The results of Corollaries 2 and 4 improve the best known algorithms for many matching problems. These are summarized in Fig. 2.

Maximum weighted graph matching (MWM) has been studied in a variety of models; the algorithm of Theorem 1 provides an approximation guarantee in any big data model where we are capable of performing greedy matching on weight-based substreams of the data. Several of these applications are explained below; each of these is an improvement of the previous best results in these models.

[3] In the case where 2α is fractional, we allow edges from OPT to be mapped "partially" to multiple edges from T so long as this doesn't result in more than 2α total OPT edges mapped to any edge of T.

Problem	Model	Previous	This Paper
MWM	One-pass streaming	4.911 [8]	4
MWM	One-pass sliding window	9.027 [6]	6
MWM	MapReduce	8 [13]	4
3-MWM	One-pass streaming	9.899 [5]	9
2-MWIS	One-pass streaming	5.828 [2]	4
3-MWIS	One-pass streaming	9.899 [2]	9

Figure 2 Approximation factor improvements over previous results. ϵ factors have been omitted.

The semi-streaming model (defined in [9]) allows one-way access to a stream of weighted edges on a machine limited to $O(n \operatorname{polylog} n)$ memory. A series of papers has provided improved approximation guarantees in this model [9, 15, 17, 8]; the current best is a $4.911 + \epsilon$ approximation [8]. Keeping a maximal matching in the semistreaming model is trivial (see e.g. [9]) and the machine has enough memory space to store one maximal matching for each of the $O(\log n)$ many weight classes, thus we find

▶ **Corollary 8.** *There is a $4 + \epsilon$ approximation algorithm for maximum weighted matching in the semistreaming model.*

Ashwinkumar 2011 [2] extended semistreaming matching algorithms to the more general case of finding maximum-weight independent sets in p-intersection systems (p-MWIS); this was further developed by Chakrabarti et al. 2013 to include weighted matching on hypergraphs of degree p (p-MWM). Our algorithm improves their approximation ratio of $2(p + \sqrt{p(p-1)}) - 1$ for the most practical $p = 2$ and $p = 3$ cases (see Figure 2).

▶ **Corollary 9.** *There is a semistreaming algorithm for finding the Maximum-Weight Independent Set on a p-system with approximation ratio $p^2 + \epsilon$.*

▶ **Corollary 10.** *There is a semistreaming algorithm for finding the Maximum-Weight Matching on a degree p hypergraph with approximation ratio $p^2 + \epsilon$.*

In the related semi-streaming "sliding window" graph model [7, 6], there is a fixed window length $L \in \omega(n \operatorname{polylog} n)$, and we are interested in maintaining (at all times) a maximum matching over the most recent L edges. We are again limited to $O(n \operatorname{polylog} n)$ memory space. In this model, only a $3 + \epsilon$ approximation to unweighted matching is known [6], and we thus find:

▶ **Corollary 11.** *There is a $6 + \epsilon$ approximation algorithm for maximum weighted matching in the semi-streaming sliding window model.*

The class $\mathcal{MRC}^0$ [12] is a theoretical model for MapReduce computations achievable with a constant number of rounds. In this model, even though the edge set does not fit on any individual processor, it is possible to find a maximal matching [13] (and thus a 2-approximation of the maximum unweighted matching). This immediately yields an improvement over the previous best-known 8-approximation algorithm for maximum weighted matching [13], with no additional communication cost (since the merge can be performed on a single processor).

▶ **Corollary 12.** *There is a constant-round $4 + \epsilon$ approximation algorithm for maximum weighted matching in the MapReduce model $\mathcal{MRC}^0$.*

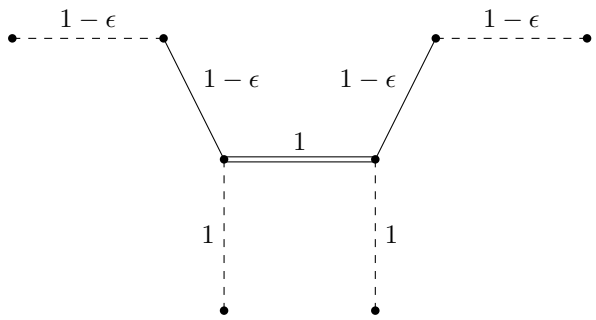


Figure 3 Graph with output weight 1 and optimum matching weight $4 - 2\epsilon$. In the weight class $[1, \infty)$ the double-lined edge is remembered; in the weight class $[1 - \epsilon, \infty)$ the two single-lined edges are remembered. The double-lined edge is output. The dashed edges are the optimum matching (and are not remembered in either class).

5 Lower Bounds for Graph Matching

In this section we consider the case of maximum weighted graph matching, and we present graph constructions which prove lower bounds on the approximation ratios achievable by our algorithm. These constructions extend to a general family of techniques which also includes previous weight-class-based approaches to maximum weighted matching.

The algorithm presented in §3 computes its output matching by performing a greedy matching on remembered edges, in decreasing order of weight. Our analysis showed that this was a $(4 + \epsilon)$-approximation. In Fig. 3 we present a graph where the algorithm's approximation ratio is $4 - 2\epsilon$, showing that our analysis is tight to within $1 + \epsilon$ factors.

In the graph of Fig. 3, the greedy matching on the remembered edges has weight 1, but the maximum weight matching on remembered edges has weight $2 - 2\epsilon$. In practice, many applications may be able to spend the post-processing time necessary to find the maximum-weight matching on the remembered edges (which are, after all, a sparse subgraph of the original graph). An obvious question is whether this post-processing can provide a stronger approximation guarantee.

The graph of Fig. 4 shows that our algorithm cannot achieve better than a 3.5 approximation, even when we output the maximum-weight matching on all of the remembered edges. Only remembered edges are drawn. Edges arrive in increasing order by weight, with the remembered edges appearing before other edges of the same weight. When the graph of Fig. 4 is extended upwards, any algorithm which uses greedy matchings on weight-based substreams cannot do better than a 3.5-approximation, because it is incapable of remembering any edge from Opt. This class of algorithms includes our algorithm and also the previous best algorithm of [8].

6 Conclusion

For specific systems of interest, we may be able to obtain stronger approximation guarantees, particularly by being more clever in our post-processing of memory. The case of one-pass streaming algorithms for graph matching is of particular interest. An obvious improvement to our algorithm is to calculate the maximum matching on all edges held in memory (via e.g. the Blossom algorithm [16]) rather than performing a greedy matching on edges held. We conjecture that this improvement yields a $(3.5 + \epsilon)$-approximation, tight to the lower bound shown in Fig. 4.

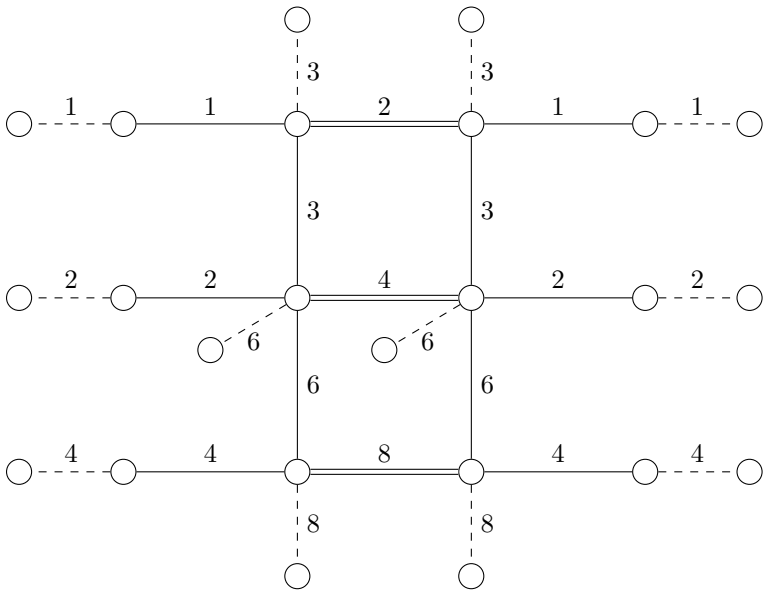


Figure 4 A graph which, when extended upwards, approaches approximation ratio 3.5. The dotted-line edges form the optimal matching, but are not remembered in any weight class. Solid edges are remembered but not output; double-lined edges are remembered and output. These remembered edges are produced by a stream where the edges arrive in order of increasing weight, with to-be-remembered edges arriving first. The reader can verify that within each weight class, the set of remembered edges is maximal.
On this graph, the output has weight 14, and the optimum matching has weight 48, for an approximation ratio of ≈3.429. The graph can be extended upwards, with each new layer including a single new output edge which decreases in weight by a power of 2; the approximation ratio quickly approaches 3.5.

Since each edge in our stream may fall in multiple weight classes, we may need to update $\Omega(\log n)$ matchings to process each incoming edge, leading to an $\Omega(\log n)$ sequential processing time. In contrast, the previous best semistreaming algorithm [8] used non-overlapping weight classes and required time $O(1)$, but obtained a worse approximation ratio. The trade-off between approximation quality and per-element processing time may be worth further study.

References

1 Alexandr Andoni, Robert Krauthgamer, and Krzysztof Onak. Streaming algorithms via precision sampling. *FOCS*, 2011.

2 B.V. Ashwinkumar. Buyback problem - approximate matroid intersection with cancellation costs. In *ICALP*, pages 379–390, 2011.

3 Ajesh Babu, Nutan Limaye, J Radhakrishnan, and Girish Varma. Streaming algorithms for language recognition problems. *Theoretical Computer Science*, 494:13–23, 2013.

4 Ziv Bar-Yossef, Ravi Kumar, and D Sivakumar. Reductions in streaming algorithms, with an application to counting triangles in graphs. In *SODA*, pages 623–632, January 2002.

5 Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: Matchings, matroids, and more. *IPCO*, 2014. To appear.

6 Michael Crouch, Andrew McGregor, and Daniel Stubbs. Dynamic graphs in the sliding-window model. *ESA*, 2013.

7 Mayur Datar, Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Maintaining stream statistics over sliding windows. *SIAM Journal on Computing*, 31(6):1794, 2002.

8 Leah Epstein, Asaf Levin, Julián Mestre, and Danny Segev. Improved approximation guarantees for weighted matching in the semi-streaming model. *SIAM Journal on Discrete Mathematics*, 25(3):1251–1265, January 2011.

9 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Theoretical Computer Science*, 348(2-3):207–216, December 2005.

10 Piotr Indyk and David P Woodruff. Optimal approximations of the frequency moments of data streams. In *STOC*, pages 202–208. ACM, 2005.

11 Thomas A Jenkyns. The efficacy of the "greedy" algorithm. *Proceedings of the 7th Southeastern Conference on Combinatorics, Graph Theory and Computing*, pages 341–350, 1976.

12 Howard J. Karloff, Siddharth Suri, and Sergei Vassilvitskii. A model of computation for MapReduce. In *SODA*, pages 938–948, 2010.

13 Silvio Lattanzi, Benjamin Moseley, Siddharth Suri, and Sergei Vassilvitskii. Filtering: a method for solving graph problems in MapReduce. In *SPAA*, New York, New York, USA, 2011. ACM Press.

14 Frédéric Magniez, Claire Mathieu, and Ashwin Nayak. Recognizing well-parenthesized expressions in the streaming model. In *STOC*, pages 261–270. ACM, 2010.

15 Andrew McGregor. Finding graph matchings in data streams. *APPROX-RANDOM*, 2005.

16 Silvio Micali and Vijay V. Vazirani. An O(sqrt(|V|) |E|) algorithm for finding maximum matching in general graphs. In *FOCS*, pages 17–27, 1980.

17 Mariano Zelke. Weighted matching in the semi-streaming model. *Algorithmica*, pages 669–680, 2012.

Guruswami-Sinop Rounding without Higher Level Lasserre

Amit Deshpande[1] **and Rakesh Venkat**[2]

1 **Microsoft Research**
Bangalore, India
amitdesh@microsoft.com

2 **Tata Institute of Fundamental Research**
Mumbai, India
rakesh@tifr.res.in

Abstract

Guruswami and Sinop [11] give a $O(1/\delta)$ approximation guarantee for the non-uniform Sparsest Cut problem by solving $O(r)$-level Lasserre semidefinite constraints, provided that the generalized eigenvalues of the Laplacians of the cost and demand graphs satisfy a certain spectral condition, namely, $\lambda_{r+1} \geq \Phi^*/(1-\delta)$. Their key idea is a rounding technique that first maps a vector-valued solution to $[0,1]$ using appropriately scaled projections onto Lasserre vectors. In this paper, we show that similar projections and analysis can be obtained using only ℓ_2^2 triangle inequality constraints. This results in a $O(r/\delta^2)$ approximation guarantee for the non-uniform Sparsest Cut problem by adding only ℓ_2^2 triangle inequality constraints to the usual semidefinite program, provided that the same spectral condition $\lambda_{r+1} \geq \Phi^*/(1-\delta)$ holds as above.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Sparsest Cut, Lasserre Hierarchy, Metric embeddings

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.105

1 Introduction

Finding sparse cuts in graphs or networks is a difficult theoretical problem with numerous practical applications, namely, divide-and-conquer graph algorithms, image segmentation [16, 17], VLSI layout [6], routing in distributed networks [5]. From the theoretical side, the problem of finding the sparsest cut in a given graph is NP-hard, and over the years, significant efforts and non-trivial ideas have gone into designing good approximation algorithms for it. The state of approximability questions for its variants such as *conductance* or *edge expansion* is also similar.

Let us first define the Sparsest Cut problem formally. The input is a pair of graphs C, D on the same vertex set V, with $|V| = n$, called the *cost* and *demand* graphs, respectively. They are specified by non-negative edge weights $c_{ij}, d_{ij} \geq 0$, for $i < j \in [n]$, and the *(non-uniform) sparsest cut* problem, henceforth referred to as Sparsest Cut, asks for a subset $S \subseteq V$ that minimizes

$$\Phi(S) = \frac{\sum_{i<j} c_{ij} \, |\mathbb{I}_S(i) - \mathbb{I}_S(j)|}{\sum_{i<j} d_{ij} \, |\mathbb{I}_S(i) - \mathbb{I}_S(j)|},$$

where $\mathbb{I}_S(i)$ is the indicator function giving 1, if $i \in S$, and 0, otherwise. We denote the optimum by $\Phi^* = \min_{S \subseteq V} \Phi(S)$. The special case of this problem where the demand graph

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 105–114

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

is a complete graph on n vertices with uniform edge weights is called the Uniform Sparsest Cut problem.

Several popular heuristics in practice for finding sparse cuts use spectral information such as the eigenvalues and eigenvectors of the underlying graph. The *generalized eigenvalues* of the Laplacian matrices of the cost and demand graphs, defined later in Section 3, provide a natural scale against which we can measure the sparsity. If $0 \leq \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_m$ are the *generalized eigenvalues* of the Laplacian matrices of cost and demand graphs, then using Courant-Fisher theorem (or the easy direction of Cheeger's inequality) we get $\lambda_1 \leq \Phi^*$. So the smallest generalized eigenvalue is at most Φ^*, and as we go to the higher eigenvalues, at some point they overtake Φ^*. We provide an approximation guarantee of

$$r \left(1 - \frac{\Phi^*}{\lambda_{r+1}}\right)^{-2}$$

for the Sparsest Cut problem, provided that $\lambda_{r+1} \geq \Phi^*$. In particular, this gives $O(r/\delta^2)$ approximation guarantee, if $\lambda_{r+1} \geq \Phi^*/(1-\delta)$. Our algorithm runs in time poly(n) and needs to solve a semidefinite program with only ℓ_2^2 triangle inequality constraints. In comparison, Guruswami-Sinop [11] give an approximation guarantee of

$$\left(1 - \frac{(1+\epsilon)\Phi^*}{\lambda_{r+1}}\right)^{-1},$$

provided that $\lambda_{r+1} \geq (1+\epsilon)\Phi^*$, but require solving a semidefinite program with $O(r/\epsilon)$ level Lasserre constraints, and hence, $2^{r/\delta\epsilon}$poly(n) running time [9].

1.1 Our Results

Our main result, proved later in Section 5, is as follows:

▶ **Theorem 1.1.** *[Main Theorem] Given an instance C, D of the* Sparsest Cut *problem, Algorithm 1 outputs a cut T that satisfies*

$$\Phi(T) \leq \min_{r \in [n]} \quad r \left(1 - \frac{\Phi^*}{\lambda_{r+1}}\right)^{-2} \Phi^*.$$

The algorithm runs in time poly(n) and needs to solve a semidefinite program with only additional ℓ_2^2 triangle inequality constraints.

Here is an immediate corollary that was mentioned in the abstract.

▶ **Corollary 1.2.** *If the input instance satisfies $\lambda_{r+1} \geq \Phi^*/(1-\delta)$ for some $r \in [n]$, then the algorithm produces a $O(r/\delta^2)$ approximation. Here, $0 \leq \lambda_1 \leq \ldots \leq \lambda_n$ are the* generalized eigenvalues *of the Laplacians of C, D.*

The proof of Theorem 1.1 is based on the following property (see Subsection 4.1) of vectors in ℓ_2^2 space that could be of independent interest.

▶ **Proposition 1.3.** *If $x_1, x_2, \ldots, x_n$ satisfy ℓ_2^2 triangle inequalities, then*

$$\left\langle x_i - x_j, \frac{x_k - x_l}{\|x_k - x_l\|} \right\rangle^2 \leq |\langle x_i - x_j, x_k - x_l \rangle| \leq \|x_i - x_j\|^2, \quad \text{for all } i, j, k, l \in [n].$$

Geometrically, this gives an embedding of $x_1, x_2, \ldots, x_n$ from ℓ_2^2 into ℓ_1 via appropriately scaled projections onto the line segment joining x_k and x_l, for any $k \neq l$. Proposition 1.3 says that this embedding is a contraction and the distortion for a pair is lower bounded by their squared distance after this projection. Thus, we can relate the average distortion to projections along certain directions.

2 Previous and Related Work

The Sparsest Cut problem has seen a lot of activity, given its central importance. For the case of Uniform Sparsest Cut, where the demand graph is the complete graph with unit demands on all pairs, the first non-trivial bound was by using Cheeger's inequality (and a corresponding algorithm)[1]. This gives an approximation factor of $1/\sqrt{\lambda_2(L)}$, where $\lambda_2(L)$ is the second-smallest eigenvalue of the normalized graph Laplacian matrix.

In in a seminal work, Leighton and Rao [15] related the problem of approximating the sparsest cut to embeddings between metric spaces, in particular, into ℓ_1. By solving a LP relaxation of the Sparsest Cut problem, they produce a metric on points and proceed to embed it into ℓ_1, and show that the worst case distortion in doing so determines the approximation factor. Using a theorem of Bourgain, they obtain an $O(\log n)$ approximation.

Following this, the breakthrough work of Arora, Rao and Vazirani [4] used a SDP (which we will refer to as ARV SDP) that could be viewed as a strengthening of both the spectral approach via Cheeger's inequality, and the distance metric approach of Leighton and Rao, to produce an $O(\sqrt{\log n})$ approximation for the Uniform Sparsest Cut. This SDP used the *triangle inequality* constraints on the squared distances between vectors crucially, and was equivalent to the problem of embedding metrics from ℓ_2^2 into ℓ_1 with low *average* distortion. Further work by Arora, Lee, and Naor [3] extended these techniques to give an $O\left(\sqrt{\log n \log\log n}\right)$ approximation for the general Sparsest Cut (equivalently, for the *worst* case distortion of ℓ_2^2 metrics into ℓ_1).

Recently, Guruswami and Sinop [12] gave a generic method for rounding a class of SDP hierarchies proposed by Lasserre [13, 14], and applied it to the Sparsest Cut problem [11]. This hierarchy subsumes the ARV SDP within 3-levels, but the size of their SDP with r levels increases as $n^{O(r)}$. The approximation guarantee depends on the *generalized eigenvalues* of the pair of Laplacians of the cost and demand graphs, and is as follows:

▶ **Theorem 2.1** (Guruswami-Sinop [11]). *Given C, D as cost and demand graphs let $0 \leq \lambda_1 \leq \lambda_2 \ldots \leq \lambda_n$ be the generalized eigenvalues between C, D. Then for every $r \in [n]$ and $\epsilon \geq 0$, a solution satisfying $O(r/\epsilon)$ levels of the Lasserre hierarchy with objective value Φ^* can be rounded to produce a cut T with value*

$$\Phi(T) \leq \Phi^* \left(1 - \frac{(1+\epsilon)\Phi^*}{\lambda_{r+1}}\right)^{-1}, \quad \text{if } \lambda_{r+1} \geq (1+\epsilon)\Phi^*.$$

For the specific case of the Uniform Sparsest Cut problem, Arora, Ge and Sinop [2] show, by using techniques from Guruswami-Sinop, that under certain conditions on the input graph (expansions of sets of size $\leq n/r$), they can get a $(1+\epsilon)$ approximation; again using the r-th level of the Lasserre hierarchy.

On the side of integrality gaps, the best known integrality gap for the ARV SDP is $(\log n)^{\Omega(1)}$ by Cheeger, Kleiner and Naor [7].

The main motivation behind this work is to get approximation guarantees similar to the Guruswami-Sinop rounding [11], but without using higher levels of the Lasserre hierarchy. Some parts of the Guruswami-Sinop proof such as column subset selection via volume sampling do not require higher level Lasserre vectors or constraints. Also the final approximation guarantee of Guruswami-Sinop does not depend on higher level Lasserre vectors. While our approximation guarantee is mildly worse than theirs, our algorithm always runs in polynomial time and does not use higher level Lasserre vectors in the rounding.

3 Notation and Preliminaries

We state the necessary notation and definitions formally in this section.

Sets, Matrices, Vectors

We use $[n] = 1, \ldots, n$. For a matrix $X \in \mathbb{R}^{d \times d}$, we say $X \succeq 0$ or X is positive-semidefinite if $y^T X y \geq 0$ for all $y \in \mathbb{R}^d$. The Gram-matrix of a matrix $M \in \mathbb{R}^{d_1 \times d_2}$ is the matrix $M^T M$, which is positive-semidefinite. We will often need the eigenvalues of the Gram-matrix of M. We will denote these by $\sigma_1(M) \geq \sigma_2(M) \geq \ldots \sigma_{d_2}(M) \geq 0$, arranged in descending order. The *Frobenius* norm of M is given by $\|M\|_F \triangleq \sqrt{\sum_i \sigma_i(M)} = \sqrt{\sum_{i \in [d_1], j \in [d_2]} M(i,j)^2}$. In our analysis, we will sometimes view a matrix M as a collection of its columns viewed as vectors; $M = (m_j)_{j \in [d_2]}$. In this case, $\|M\|_F^2 = \sum_j \|m_j\|^2$.

Generalized Eigenvalues

Given two symmetric matrices $X, Y \in \mathbb{R}^d \times d$ with $Y \succeq 0$, and for $i \leq \text{rank}(Y)$, we define their i-th smallest generalized eigenvalue as the following:

$$\lambda_i = \max_{\text{rank}(Z) \leq i-1} \quad \min_{w \perp Z; w \neq 0} \quad \frac{w^T X w}{w^T Y w}$$

Graphs and Laplacians

All graphs will be defined on a vertex set V of size n. The vertices will usually be referred to by indices $i, j, k, l \in [n]$. Given a graph with weights on pairs $W : \binom{V}{2} \mapsto \mathbb{R}^+$, the graph Laplacian matrix is defined as:

$$L_W(i,j) = \begin{cases} -W(i,j) & \text{if } i \neq j \\ \sum_k W(i,k) & \text{if } i = j \end{cases}$$

Sparsest Cut SDP

The SDP we use for Sparsest Cut on the vertex set V with costs and demands $c_{ij}, d_{kl} \geq 0$ and corresponding cost and demand graphs $C : \binom{V}{2} \mapsto \mathbb{R}^+$ and $D : \binom{V}{2} \mapsto \mathbb{R}^+$, is effectively the following:

$$\textbf{SDP:} \quad \Phi(SDP) = \min \frac{\sum_{i<j} c_{ij} \|x_i - x_j\|^2}{\sum_{k<l} d_{kl} \|x_k - x_l\|^2} \tag{1}$$

$$\text{subject to} \quad \|x_i - x_j\|^2 + \|x_j - x_k\|^2 \geq \|x_i - x_k\|^2 \quad \forall i, j, k \in [n] \tag{2}$$

While this is technically not an SDP due to the presence of a fraction in the objective, it is not difficult to see that we can construct an equivalent SDP as shown in [11]. We will use $\Phi(ALG)$ to denote the sparsity of the cut produced by an algorithm, and will compare it to $\Phi(SDP)$. Note that any set of vectors $x_1, \ldots, x_n$ that are feasible for this SDP satisfy the triangle inequalities on the *squares* of their distances, and are said to satisfy the ℓ_2^2 triangle inequality, or are in ℓ_2^2 space.

Lasserre Hierarchy

The Lasserre hierarchy [13] at level r strengthens the basic SDP relaxation by introducing new vectors, $x_S(f)$, for every $S \subseteq [n]$ with $|S| \leq r$ and every $f : S \to \{0,1\}^{|S|}$, and requiring certain consistency conditions on the inner products between them. We do not go into the details of the hierarchy here, since we will not be using it in this work. We refer the reader to available surveys, e.g. [14] for more details. For the Sparsest Cut problem, one can show that the ℓ_2^2 triangle inequalities are subsumed by 3 levels of this hierarchy.

ℓ_1 embeddings and cuts

Leighton and Rao [15] show that instead of producing cuts, it is sufficient to produce a mapping $Z : V \to \mathbb{R}^d$, with $z_i = Z(i)$, from which we can extract a cut T such that

$$\Phi(T) \leq \frac{\sum_{i<j} c_{ij} \|z_i - z_j\|_1}{\sum_{k<l} d_{kl} \|z_k - z_l\|_1}.$$

This follows from the fact that ℓ_1 metrics are exactly the cone of cut-metrics.

4 Lasserre hierarchy vs. ℓ_2^2 triangle inequality

Let's first recap Guruswami-Sinop [11, 12, 10, 9] to demonstrate its key ideas and to facilitate its comparison with our method coming later. At the basic level, they map SDP solution vectors to values in $[0,1]$, where one can then run independent or threshold rounding. To define this map, they need $O(r)$-level Lasserre vectors $\{x_S(f)\}_{S,f}$ for subsets $S \subseteq [n]$ of size at most $O(r)$ and assignments $f \in \{0,1\}^{|S|}$. For simplicity of notation, call $x_{\{i\}}(1)$ as x_i. Now the algorithm has two parts.

1. Pick a subset S of size $O(r)$ using volume sampling [8] on the matrix with columns as $\{\sqrt{d_{ij}}(x_i - x_j)\}_{i<j}$. This part does not require Lasserre vectors or constraints in the algorithm as well as the analysis.
2. For the S fixed as above, pick $x_S(f)$ with probability $\propto \|x_S(f)\|^2$ and map each x_i to $p_i^{(f)} \in [0,1]$ as follows.

$$x_i \mapsto p_i^{(f)} = \frac{\langle x_i, x_S(f)\rangle}{\|x_S(f)\|^2} \in [0,1].$$

Once we have $p_i^{(f)} \in [0,1]$ for all $i \in [n]$, we can either do threshold rounding with a random threshold $r \in [0,1]$ or do independent rounding with $p_i^{(f)}$'s as probabilities. Lasserre constraints are used to show $p_i^{(f)} \in [0,1]$ and the following important property used in the analysis.

$$\left\langle x_i - x_j, \frac{x_S(f)}{\|x_S(f)\|} \right\rangle^2 \leq |\langle x_i - x_j, x_S(f)\rangle| \leq \|x_i - x_j\|^2, \quad \text{for all } i,j \in [n].$$

What is special about these directions $x_S(f)$? Are there other directions that exhibit similar property and can be found without solving multiple levels of Lasserre hierarchy?

4.1 ℓ_2^2 triangle inequality

We make an interesting observation that ℓ_2^2 triangle inequalities give a large collection of vectors that exhibit the same property as the $x_S(f)$'s used in the analysis of Guruswami-Sinop.

ℓ_2^2 triangle inequalities for all triplets, or equivalently, the acuteness of all angles in a point set $\{x_1, x_2, \ldots, x_n\}$ can be written as $\langle x_i - x_l, x_k - x_l \rangle \geq 0$, for all $i, k, l \in [n]$, and gives the following interesting mapping of vectors x_i to values $p_i^{(k,l)} \in [0,1]$ as

$$x_i \mapsto p_i^{(k,l)} = \frac{\langle x_i - x_l, x_k - x_l \rangle}{\|x_k - x_l\|^2}.$$

Note that $p_i^{(k,l)}$ depends on the ordered pair (k,l), and $p_i^{(k,l)} \in [0,1]$ by the ℓ_2^2 triangle inequalities or acuteness of all angles. Another interesting consequence is

$$1 - p_i^{(k,l)} = \frac{\langle x_k - x_i, x_k - x_l \rangle}{\|x_k - x_l\|^2}.$$

Moreover, we show that the direction $x_k - x_l$ behaves similar to $x_S(f)$ used in the analysis of Guruswami-Sinop.

▶ **Proposition 4.1.** *[Restatement of Proposition 1.3] If $x_1, x_2, \ldots, x_n$ satisfy ℓ_2^2 triangle inequalities, then*

$$\left\langle x_i - x_j, \frac{x_k - x_l}{\|x_k - x_l\|} \right\rangle^2 \leq |\langle x_i - x_j, x_k - x_l \rangle| \leq \|x_i - x_j\|^2, \quad \textit{for all } i, j, k, l \in [n].$$

Proof. By acuteness of all angles, we know that

$$\langle x_i - x_k, x_i - x_j \rangle \geq 0 \quad \text{and} \quad \langle x_l - x_j, x_i - x_j \rangle \geq 0, \quad \text{for all } i, j, k, l \in [n].$$

Adding both the inequalities we get $\|x_i - x_j\|^2 - \langle x_k - x_l, x_i - x_j \rangle \geq 0$, or equivalently $\langle x_k - x_l, x_i - x_j \rangle \leq \|x_i - x_j\|^2$. Since swapping k and l does not affect the above argument, we get the upper bound

$$|\langle x_k - x_l, x_i - x_j \rangle| \leq \|x_i - x_j\|^2, \quad \text{for all } i, j, k, l \in [n].$$

Swapping (i,j) and (k,l), we also have $|\langle x_k - x_l, x_i - x_j \rangle| \leq \|x_k - x_l\|^2$. Therefore,

$$\begin{aligned} \left\langle x_i - x_j, \frac{x_k - x_l}{\|x_k - x_l\|} \right\rangle^2 &= \frac{\langle x_k - x_l, x_i - x_j \rangle^2}{\|x_k - x_l\|^2} \\ &\leq \frac{\langle x_k - x_l, x_i - x_j \rangle^2}{|\langle x_k - x_l, x_i - x_j \rangle|} \\ &= |\langle x_k - x_l, x_i - x_j \rangle|. \end{aligned}$$

◀

4.2 Low dimensional SDP solutions

Although the Guruswami-Sinop [11] result is finally stated in terms of a condition on generalized eigenvalues, it can also be thought of as a result that gives good approximation guarantees when the SDP solution is close to being low rank. Suppose the Gram matrix of $\{x_i - x_j\}_{1 \leq i < j \leq n}$ has at least δ fraction of its spectrum in its top r eigenvalues, that is, $\sum_{t=1}^{r} \lambda_t \geq \delta \sum_{t=1}^{n} \lambda_t$, where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n \geq 0$ are the eigenvalues of the Gram matrix of $\{x_i - x_j\}_{1 \leq i < j \leq n}$. Then Proposition 4.2 proves the existence of a good direction $x_k - x_l$ by weighted averaging.

▶ **Proposition 4.2.** *If $x_1, x_2, \ldots, x_n$ satisfy the above spectral or low-rank property, then there exists $x_k - x_l$ such that*

$$\sum_{i<j} \left\langle x_i - x_j, \frac{x_k - x_l}{\|x_k - x_l\|} \right\rangle^2 \geq \frac{\delta^2}{r} \sum_{i<j} \|x_i - x_j\|^2 .$$

Proof. To show the existence of a good $x_k - x_l$, we take expectation over $x_k - x_l$ by squared length sampling.

$$\begin{aligned}
\max_{k<l} \sum_{i<j} \left\langle x_i - x_j, \frac{x_k - x_l}{\|x_k - x_l\|} \right\rangle^2 &\geq \sum_{k<l} \frac{\|x_k - x_l\|^2}{\sum_{p<q} \|x_p - x_q\|^2} \sum_{i<j} \left\langle x_i - x_j, \frac{x_k - x_l}{\|x_k - x_l\|} \right\rangle^2 \\
&= \frac{\sum_{k<l} \sum_{i<j} \langle x_i - x_j, x_k - x_l \rangle^2}{\sum_{p<q} \|x_p - x_q\|^2} \\
&= \frac{\sum_{t=1}^n \lambda_t^2}{\sum_{t=1}^n \lambda_t} \\
&\geq \frac{\sum_{t=1}^r \lambda_t^2}{\sum_{t=1}^n \lambda_t} \\
&\geq \frac{\left(\sum_{t=1}^r \lambda_t\right)^2}{r \sum_{t=1}^n \lambda_t} \qquad \text{by Cauchy-Schwarz inequality} \\
&\geq \frac{\delta^2}{r} \sum_{t=1}^n \lambda_t \qquad \text{by the spectral or low-rank property} \\
&= \frac{\delta^2}{r} \sum_{i<j} \|x_i - x_j\|^2 .
\end{aligned}$$

◀

5 Non-uniform sparsest cut

We now give the proof of the Main Theorem (Theorem 1.1). The rounding algorithm is Algorithm 1.

Algorithm 1 Algorithm for Sparsest Cut

Input: C, D and a solution $\{x_1, \ldots, x_n\}$ to the ARV SDP for Sparsest Cut
Output: A cut $(T, \bar{T})$

1: **for all** Pairs $(k, l) \in [n] \times [n]$ **do**
2: $\quad p_i^{(k,l)} = \dfrac{\langle x_i - x_l, x_k - x_l \rangle}{\|x_k - x_l\|^2}$ `% line embedding`
3: $\quad$ **for all** $t \in [n]$ **do**
4: $\quad\quad S_{kl}^{(t)} = \left\{ i \ : \ p_i^{(k,l)} \leq p_t^{(k,l)} \right\}$ `% threshold rounding`
5: $\quad$ **end for**
6: **end for**
7: $T = \arg\min_{k,l,t} \Phi\left(S_{kl}^{(t)}\right)$
8: Output the cut $(T, \bar{T})$

Algorithm 1 goes over all directions $x_k - x_l$. For each of them, it maps x_i to $p_i \in [0, 1]$ as

$$x_i \mapsto p_i^{(k,l)} = \frac{\langle x_i - x_l, x_k - x_l \rangle}{\|x_k - x_l\|^2}.$$

Now for each $t \in [n]$ consider the sweep cut $S_t = \{j \ : \ p_j^{(k,l)} \leq p_t^{(k,l)}\}$, and output the best amongst them as T.

For convenience of notation, we will do the analysis using the corresponding ℓ_1 embedding, as mentioned in Section 3. Given an ℓ_1-embedding, we can get a cut with similar guarantee by choosing the best threshold cut along each coordinate, which is what our algorithm does. Define an ℓ_1-embedding of x_i's as follows.

$$x_i \mapsto y_i = \left(\frac{d_{kl} \left\| x_k - x_l \right\|^2 \langle x_i - x_l, x_k - x_l \rangle}{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2} \right)_{k<l}.$$

The following is an easy consequence of Proposition 4.1.

▶ **Proposition 5.1.**

$$\frac{\sum_{k<l} d_{kl} \langle x_i - x_j, x_k - x_l \rangle^2}{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2} \leq \left\| y_i - y_j \right\|_1 \leq \left\| x_i - x_j \right\|^2,$$

Proof. Let's start with the upper bound.

$$\begin{aligned} \left\| y_i - y_j \right\|_1 &= \frac{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2 |\langle x_i - x_j, x_k - x_l \rangle|}{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2} \\ &\leq \frac{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2 \left\| x_i - x_j \right\|^2}{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2} \qquad \text{by Proposition 4.1} \\ &= \left\| x_i - x_j \right\|^2. \end{aligned}$$

Now the lower bound.

$$\begin{aligned} \left\| y_i - y_j \right\|_1 &= \frac{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2 |\langle x_i - x_j, x_k - x_l \rangle|}{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2} \\ &\geq \frac{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2 \left\langle x_i - x_j, \frac{x_k - x_l}{\|x_k - x_l\|} \right\rangle^2}{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2} \qquad \text{by Proposition 4.1} \\ &= \frac{\sum_{k<l} d_{kl} \langle x_i - x_j, x_k - x_l \rangle^2}{\sum_{k<l} d_{kl} \left\| x_k - x_l \right\|^2}. \end{aligned}$$

◀

Equipped with this, we can now bound the average distortion, and hence, the approximation factor of our algorithm. We use the following Proposition from Guruswami-Sinop [11] to rewrite the final bound in terms of the generalized eigenvalues of the Laplacian matrices of the cost and demand graphs.

▶ **Proposition 5.2.** *[11] Let $0 \leq \lambda_1 \leq \ldots \leq \lambda_m$ be the generalized eigenvalues of the Laplacian matrices of the cost and demand graphs. Let $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0$ be eigenvalues of the Gram matrix of vectors $\{\sqrt{d_{ij}}(x_i - x_j)\}_{i<j}$. Then*

$$\frac{\sum_{t \geq r+1} \sigma_j}{\sum_{t=1}^{n} \sigma_j} \leq \frac{\Phi(SDP)}{\lambda_{r+1}}.$$

Using these we bound the approximation ratio of our algorithm and prove Theorem 1.1.

▶ **Theorem 5.3.** *[Restatement of Theorem 1.1]*

$$\Phi(ALG) \leq \Phi(SDP) \cdot r \left(1 - \frac{\Phi(SDP)}{\lambda_{r+1}}\right)^{-2}.$$

Proof. The guarantee of our algorithm can only be better than the guarantee of this corresponding ℓ_1-embedding.

$$\begin{aligned}
\Phi(ALG) &\leq \frac{\sum_{i<j} c_{ij} \|y_i - y_j\|_1}{\sum_{i<j} d_{ij} \|y_i - y_j\|_1} \\
&\leq \frac{\sum_{i<j} c_{ij} \|x_i - x_j\|^2 \sum_{k<l} d_{kl} \|x_k - x_l\|^2}{\sum_{i<j} d_{ij} \sum_{k<l} d_{kl} \langle x_i - x_j, x_k - x_l \rangle^2} \\
&= \frac{\sum_{i<j} c_{ij} \|x_i - x_j\|^2}{\sum_{i<j} d_{ij} \|x_i - x_j\|^2} \cdot \frac{\left(\sum_{i<j} d_{ij} \|x_i - x_j\|^2\right) \left(\sum_{k<l} d_{kl} \|x_k - x_l\|^2\right)}{\sum_{i<j} \sum_{k<l} d_{ij} d_{kl} \langle x_i - x_j, x_k - x_l \rangle^2} \\
&= \Phi(SDP) \cdot \frac{\left(\sum_{i<j} d_{ij} \|x_i - x_j\|^2\right)^2}{\sum_{i<j} \sum_{k<l} d_{ij} d_{kl} \langle x_i - x_j, x_k - x_l \rangle^2} \\
&= \Phi(SDP) \cdot \frac{\left(\sum_{t=1}^n \sigma_t\right)^2}{\sum_{t=1}^n \sigma_t^2} \\
&\leq \Phi(SDP) \cdot \frac{\left(\sum_{t=1}^n \sigma_t\right)^2}{\sum_{t=1}^r \sigma_t^2} \\
&\leq \Phi(SDP) \cdot r \left(\frac{\sum_{t=1}^n \sigma_t}{\sum_{t=1}^r \sigma_t}\right)^2 \qquad \text{by Cauchy-Schwarz inequality} \\
&\leq \Phi(SDP) \cdot r \left(1 - \frac{\sum_{t \geq r+1} \sigma_t}{\sum_{t=1}^n \sigma_t}\right)^{-2} \\
&\leq \Phi(SDP) \cdot r \left(1 - \frac{\Phi(SDP)}{\lambda_{r+1}}\right)^{-2} \qquad \text{by Proposition 5.2} \\
&\leq \Phi^* \cdot r \left(1 - \frac{\Phi^*}{\lambda_{r+1}}\right)^{-2}.
\end{aligned}$$

◀

6 Conclusion

We show that it is possible to get approximation guarantees similar to Guruswami-Sinop for the Sparsest Cut problem, but without using higher level Lasserre vectors. One obvious question that arises out of this is whether we can apply these techniques with threshold or independent rounding to give similar guarantees for other problems. Further, can we obtain more directions for projections and sweep cuts using lower levels of the Lasserre hierarchy or eigenvectors of the SDP solution?

Acknowledgement. The authors would like to thank Prahladh Harsha for many valuable discussions.

References

1 N. Alon and V. D. Milman. λ_1, isoperimetric inequalities for graphs, and superconcentrators. *Journal of Combinatorial Theory, Series B*, 38(1):73–88, 1985.

2 Sanjeev Arora, Rong Ge, and Ali Kemal Sinop. Towards a Better Approximation for Sparsest Cut? In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 270–279, Los Alamitos, CA, USA, 2013. IEEE Computer Society.

3 Sanjeev Arora, James R. Lee, and Assaf Naor. Euclidean distortion and the Sparsest Cut. In *In Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 553–562. ACM Press, 2005.

4 Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander Flows, Geometric Embeddings and Graph Partitioning. *J. ACM*, 56(2):5:1–5:37, April 2009.

5 Baruch Awerbuch and David Peleg. Sparse partitions (extended abstract). In *FOCS*, pages 503–513, 1990.

6 Sandeep N. Bhatt and Frank Thomson Leighton. A Framework for Solving VLSI Graph Layout Problems. *J. Comput. Syst. Sci.*, 28(2):300–343, 1984.

7 Jeff Cheeger, Bruce Kleiner, and Assaf Naor. A $(\log n)^{\Omega(1)}$ Integrality Gap for the Sparsest Cut SDP. In *FOCS*, pages 555–564, 2009.

8 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.

9 Venkatesan Guruswami and Ali Kemal Sinop. Faster SDP Hierarchy Solvers for Local Rounding Algorithms. In *FOCS*, pages 197–206, 2012.

10 Venkatesan Guruswami and Ali Kemal Sinop. Optimal column-based low-rank matrix reconstruction. In *SODA*, pages 1207–1214, 2012.

11 Venkatesan Guruswami and Ali Kemal Sinop. Approximating Non-Uniform Sparsest Cut Via Generalized Spectra. In *SODA*, pages 295–305, 2013.

12 Venkatesan Guruswami and Ali Kemal Sinop. Rounding Lasserre SDPs using column selection and spectrum-based approximation schemes for graph partitioning and Quadratic IPs. *CoRR*, abs/1312.3024, 2013.

13 Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11:796–817, 2001.

14 Monique Laurent. A Comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre Relaxations for 0–1 Programming. *Mathematics of Operations Research*, 28(3):470–496, 2003.

15 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.

16 Jianbo Shi and Jitendra Malik. Normalized Cuts and Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

17 Ali Kemal Sinop and Leo Grady. Uninitialized, globally optimal, graph-based rectilinear shape segmentation the opposing metrics method. In *ICCV*, pages 1–8, 2007.

Improved Approximation Algorithm for Steiner k-Forest with Nearly Uniform Weights

Michael Dinitz[1], Guy Kortsarz*[2], and Zeev Nutov[3]

1 **Johns Hopkins University**
`mdinitz@cs.jhu.edu`
2 **Rutgers University, Camden**
`guyk@camden.rutgers.edu`
2 **The Open University of Israel**
`nutov@openu.ac.il`

Abstract

In the Steiner k-Forest problem we are given an edge weighted graph, a collection D of node pairs, and an integer $k \leq |D|$. The goal is to find a minimum cost subgraph that connects at least k pairs. The best known ratio for this problem is $\min\{O(\sqrt{n}), O(\sqrt{k})\}$ [8]. In [8] it is also shown that ratio ρ for Steiner k-Forest implies ratio $O(\rho \cdot \log^2 n)$ for the Dial-a-Ride problem: given an edge weighted graph and a set of items with a source and a destination each, find a minimum length tour to move each object from its source to destination, but carrying at most k objects at a time. The only other algorithm known for Dial-a-Ride, besides the one resulting from [8], has ratio $O(\sqrt{n})$ [4]. We obtain ratio $n^{0.448}$ for Steiner k-Forest and Dial-a-Ride with unit weights, breaking the $O(\sqrt{n})$ ratio barrier for this natural special case. We also show that if the maximum weight of an edge is $O(n^{\epsilon})$, then one can achieve ratio $O(n^{(1+\epsilon)\cdot 0.448})$, which is less than $\sqrt{n}$ if ϵ is small enough. To prove our main result we consider the following generalization of the Minimum k-Edge Subgraph (Mk-ES) problem, which we call Min-Cost ℓ-Edge-Profit Subgraph (MCℓ-EPS): Given a graph $G = (V, E)$ with edge-profits $p = \{p_e : e \in E\}$ and node-costs $c = \{c_v : v \in V\}$, and a lower profit bound ℓ, find a minimum node-cost subgraph of G of edge profit at least ℓ. The Mk-ES problem is a special case of MCℓ-EPS with unit node costs and unit edge profits. The currently best known ratio for Mk-ES is $n^{3-2\sqrt{2}+\epsilon}$ (note that $3 - 2\sqrt{2} < 0.1716$) [5]. We extend this ratio to MCℓ-EPS for arbitrary node weights and edge profits that are polynomial in n, which may be of independent interest.

1998 ACM Subject Classification G.2.2 Graph Theory: Graph algorithms

Keywords and phrases k-Steiner Forest, Uniform weights, Densest k-Subgraph, Approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.115

1 Introduction

We consider the following problem, originally introduced by Hajiaghayi and Jain [9]:

Steiner k-Forest
Instance: A graph $G = (V, E)$ with edge-weights $w = \{w_e : e \in E\}$, a collection D of node pairs (called the *demand pairs*), and an integer $k \leq |D|$.
Objective: Find a minimum weight subgraph of G that connects at least k pairs from D.

* Partially supported by NSF award number 1218620.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 115–127

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

Steiner k-Forest generalizes several well known problems, among them the following:

- For $k = |D|$ we get the Steiner Forest problem, which admits a 2-approximation algorithm [1].
- When the demand pairs form a star (namely, when there is a node that belongs to all the demand pairs) and $|D| = n - 1$ we get the k-MST problem, which admits a 2-approximation algorithm [7].
- When $k = |D|$ and the demand pairs form a star we get the Steiner Tree problem, which admits a $(\ln 4 + \epsilon)$-approximation scheme [3].
- When G contains a spanning star and all edges have unit weights, we get the Minimum k-Edge Subgraph (Mk-ES) problem: given a graph G and an integer k, find a subgraph of G with k edges and minimum number of nodes (see [9] for the reduction details). This problem admits an $n^{3-2\sqrt{2}+\epsilon}$-approximation scheme [5] (note that $3 - 2\sqrt{2} < 0.1716$).

The best known ratio for Steiner k-Forest is $\min\{O(\sqrt{n}), O(\sqrt{k})\}$ [8], even for the case of unit weights. For $k = O(n)$ this ratio is essentially no better than the best known ratio $k^{1/2+\epsilon}$ for the directed version of the problem [6], even though undirected network design problems are usually much easier to approximate than their directed variants. We prove the following.

▶ **Theorem 1.** Steiner k-Forest *with unit weights admits an* $n^{\frac{1}{3}(7-4\sqrt{2})+\epsilon}$*-approximation scheme.*

Note that $\frac{1}{3}(7 - 4\sqrt{2}) < 0.44772$, so this is a polynomial improvement over the previous $O(\sqrt{n})$-approximation.

To prove Theorem 1 we consider the following generalization of the Mk-ES problem, which we call Min-Cost ℓ-Edge-Profit Subgraph, or MCℓ-EPS for short.

Min-Cost ℓ-Edge-Profit Subgraph (MCℓ-EPS)
Instance: A graph $G = (V, E)$ with edge-profits $p = \{p_e : e \in E\}$ and node-costs $c = \{c_v : v \in V\}$, and a profit lower bound ℓ.
Objective: Find a minimum node-cost subgraph of G of profit at least ℓ.

MCℓ-EPS with unit node costs and unit edge profits (and $\ell = k$) is the Mk-ES problem. As was mentioned, the currently best known ratio for Mk-ES is $n^{3-2\sqrt{2}+\epsilon}$ [5]. We extend this ratio to MCℓ-EPS by modifying the algorithm of [5] to handle weights and profits (essentially by adding an extra preprocessing step). Our extension can handle general node weights and profits bounded by a polynomial in n. Thus the node costs can be exponential in n or beyond. When the edge profits are exponential in n we can only give a bicriteria approximation: the algorithm will find a subgraph in which the total node weight is at most $n^{3-2\sqrt{2}+\epsilon}$ worse than the optimum, but it only covers edges with at least $\ell(1 - 1/poly(n))$ profit rather than the desired profit of ℓ (where $poly(n)$ is any polynomial in n). However, in the application to Steiner k-Forest edge profits are at most n^2, and hence we do not have to resort to the bicriteria approximation.

▶ **Theorem 2.** MCℓ-EPS *with edge profits that are at most polynomial in* n *(but with arbitrary node costs) admits an* $n^{3-2\sqrt{2}+\epsilon}$*-approximation scheme.*

The following theorem establishes a relation between Steiner k-Forest and MCℓ-EPS, and it implies Theorem 1 by substituting the value of $\gamma = 3 - 2\sqrt{2} + \epsilon$ from Theorem 2.

▶ **Theorem 3.** *If* MCℓ-EPS *admits approximation ratio* $\rho = n^{\gamma}$, $0 \le \gamma \le 1/4$, *then* Steiner k-Forest *with unit weights admits approximation ratio* $\tilde{O}\left(n^{1/3+2\gamma/3}\right)$.

This theorem forms our core technical contribution, and most of the rest of the paper is devoted to proving it.

Another problem closely related to Steiner k-Forest is the following:

Dial-a-Ride
Instance: A graph $G = (V, E)$ with edge-lengths $w = \{w_e : e \in E\}$, a collection of items with a source and a destination each, and an integer k.
Objective: Move every item from its source to its destination using a vehicle that can carry at most k items, minimizing total travel length.

Charikar and Raghavachari [4] showed that this problem admits ratio $O(\sqrt{n})$, while Gupta et al. [8] showed that ratio ρ for Steiner k-Forest implies ratio $O(\rho \cdot \log^2 n)$ for Dial-a-Ride. Note that in Theorem 9 of [8], the Dial-a-Ride problem is approximated using the approximation for Steiner k-Forest as a *black box*. Thus if the Dial-a-Ride problem has uniform edge costs, the black box can be replaced by our approximation for Steiner k-Forest. This implies the same approximation (up to $polylog(n)$ factors) for uniform edge cost Dial-a-Ride.

▶ **Corollary 4.** *There is an* $n^{\frac{1}{3}(7-4\sqrt{2})+\epsilon}$*-approximation scheme for* Dial-a-Ride *with unit edge weights.*

We note that the unit weight versions of Steiner k-Forest and Dial-a-Ride are natural special cases to consider. In addition, the Mk-ES problem is a special case of Steiner k-Forest with unit weights.

It is easy to see (and is a relatively standard observation) that the input to Steiner k-Forest can be replaced by a graph spanner with $O(\log n)$ stretch and $O(n)$ edges, while only paying an extra $O(\log n)$ in the approximation ratio [2]. Thus as long as the average edge weight in the spanner is at most n^{ϵ} we can obtain an $O(n^{(1+\epsilon)0.448})$ approximation ratio by replacing every edge by a path. This ratio is better than $\sqrt{n}$ if ϵ is small enough. This is true, for example, if the maximum cost of an edge in the original graph is at most $O(n^{\delta})$ for small enough δ.

Our techniques may be the first step towards breaking the $O(\sqrt{n})$ barrier for both Steiner k-Forest and Dial-a-Ride with general edge weights. If this is not possible, then we get the somewhat rare case in which the unweighted version of a network design problem admits an approximation ratio that is better by a polynomial factor than what is possible in the weighted case.

2 A High Level Overview of the Main Ideas

We now turn to proving Theorem 3, which as discussed suffices to prove Theorem 1. Since it is rather involved, in this section we give a quick outline of the main ideas. We assume that we know the optimal solution value, which we denote by τ. This is without loss of generality, since we can just run our algorithm on every possible τ.

At a high level, our algorithm is a two-step process. In the first step we find a set of trees which together contain k demand pairs, and in the second step we connect *all* pairs of terminals in different trees. In order to bound the cost of the second step, we will make sure that we use very few trees, and that the distance between any two trees is at most τ. This will let us connect the trees very cheaply, since we can just arbitrarily connect them together and pay at most the number of trees times τ, losing us at most the number of trees in our approximation ratio.

So the problem boils down to finding a small set of trees that together contain many demand pairs, and which are of pairwise distance at most τ. We first build a set of candidate trees, and from them select a small set (possibly with some modifications to the trees). A cluster decomposition is a collection of clusters, and a cluster is a collection $\mathcal{T}$ of trees with some particular properties. We will guarantee that every terminal belongs to exactly one tree of exactly one cluster. Moreover, trees in the same cluster are node-disjoint (not just terminal-disjoint). Each tree has diameter at most $2d$ for some parameter d, and the distance between every two terminals in different trees of the same cluster is at least $2d$.

While the trees from the cluster decomposition cover the demand, there is no upper bound on the number of trees or on the cost of any particular tree. We divide the trees into light trees (those with few edges compared to τ) and heavy trees (those with many edges). If we choose to take a light tree we can simply take the entire tree, since it is small by definition. But heavy trees have to be handled differently: we might choose to take part of a heavy tree, rather than the entire thing.

It turns out because of how we construct the clusters, we can prove that at most τ/d trees from any cluster can intersect the optimum solution, and that there are always two clusters which together contain a significant amount of demand pairs. This means that there is a way of covering at least k demand pairs using few trees, since the optimum solution does. Thus we can use the algorithm we develop for MCℓ-EPS to select few trees which together contain at least k demand pairs. This is fine if they are light trees, but if they are heavy trees we instead must select a subset of terminals in the tree to actually connect without using too many edges. Fortunately this can be handled simultaneously using our MCℓ-EPS algorithm.

3 Proof of Theorem 3

Fix some optimal solution J and a set D_J of k demand pairs connected by J. We will use the following notation.

- $\tau = |J|$ is the optimum solution value, namely, the number of edges in J.
- R_J is the union of the pairs in D_J, and $q = |R_J|$ is the number of nodes in R_J.
- Recall that $\rho = n^\gamma$ denotes the best known ratio for MCℓ-EPS.

In what follows, we may "guess" the right values of τ and q, by applying any of our algorithms for all possible values of $\tau = 1, \ldots, |E|$ and $q = 1, \ldots, n$, and among the edge sets computed return the best one. Note that $q = |R_J| \leq 2|J| \leq 2\tau$, since every node in R_J is an endnode of some edge in J.

We first give some bounds on q and τ by showing easy algorithms for special cases.

▶ **Lemma 5.** *For any $0 \leq \theta \leq 1/2$ the following holds: unless $n^{1/2-\theta} \leq q \leq 2\tau \leq n^{1/2+\theta}$,* Steiner k-Forest *with unit weights admits approximation ratio $O(n^{1/2-\theta})$.*

Proof. Any maximal forest of G is a feasible solution that has at most $n-1$ edges, so if $\tau \geq n^{1/2+\theta}$ then simply returning a maximal forest guarantees an approximation ratio $(n-1)/n^{1/2+\theta} < n^{1/2-\theta}$. If $q < n^{1/2-\theta}$, then we claim that the ratio $O(n^{1/2-\theta})$ follows from the ratio $O(\sqrt{k})$ of [8]. This is since $k \leq q(q-1)/2$, and thus $\sqrt{k} \leq q < n^{1/2-\theta}$. ◀

▶ **Corollary 6.** *For any $0 \leq \gamma \leq 1/4$, the following holds: if $\tau\sqrt{q} > n^{1-\gamma}$, then* Steiner k-Forest *with unit weights admits approximation ratio $O\left(n^{1/3+2\gamma/3}\right)$.*

Proof. Let $\theta = \frac{1}{6} - \frac{2}{3}\gamma$. Since $\gamma \leq \frac{1}{4}$, $\theta \geq 0$. We claim that $\tau > n^{1/2+\theta}$, and then ratio $O\left(n^{1/2-\theta}\right) = O\left(n^{1/3+2\gamma/3}\right)$ follows from Lemma 5. To see this, note that if $\tau \leq n^{1/2+\theta}$ then also $q \leq n^{1/2+\theta}$, so $\tau\sqrt{q} \leq n^{1/2+\theta}n^{1/4+\theta/2} = n^{3/4+3\theta/2} = n^{1-\gamma}$, which is a contradiction. ◀

The next lemma forms the technical heart of our paper, and is proved in Section 4. It says that we can do step 1 from the overview: there is a way of finding a subgraph (i.e. the collection of trees discussed in Section 2) that is relatively cheap, contains few components, and contains a lot of demand pairs.

▶ **Lemma 7.** *There exists a polynomial time algorithm that when given a* Steiner k-Forest *instance and integers $1 \leq d, h \leq n/2$, computes a subgraph $G' = (V', E')$ of G such that the following holds: G' has $\tilde{O}(\rho\tau/d + n/h)$ connected components, V' contains $\tilde{\Omega}(k)$ pairs from D, and $|E'| = \tilde{O}(\rho h\tau/d + \rho qd)$.*

We now want to give an algorithm that uses this lemma. We first need an easy lemma: an algorithm for Steiner Forest where we bound the total cost by the number of nodes and the maximum distance.

▶ **Lemma 8.** Steiner Forest *(with arbitrary weights) admits a polynomial time algorithm that computes a solution F' of size at most $L(|R| - 1)$, where $L = \max_{uv \in D} \text{dist}_G(u, v)$ and R is the union of the demand pairs.*

Proof. Let (V, D') be a spanning forest of the demand graph (V, D). The connected components of (V, D') coincide with those of (V, D). For every $\{u, v\} \in D'$ let P_{uv} be the edge set of a shortest uv-path, and let F' be the union of these edge sets. It is easy to see that the graph (V, F) connects every pair in D, and clearly its weight is at most $L(|R| - 1)$. ◀

We can now give our algorithm: we simply connect each of the components we are given to each other by using the above Steiner Forest algorithm. We write this a little more formally as Algorithm 1.

Algorithm 1: PARTIAL-CONNECT(G, D, τ, h, d)

1. Remove from D every pair u, v with $\text{dist}_G(u, v) > \tau$.
2. Compute a graph $G' = (V', E')$ using Lemma 7.
3. Contract every connected component of G' into a single node and update the set of demand pairs accordingly. For the obtained instance of Steiner Forest, compute an edge set F' as in Lemma 8.
4. **return** $E' \cup F'$

▶ **Theorem 9.** *Given integers $1 \leq d, h \leq n/2$, Algorithm 1 returns a graph that covers $\tilde{\Omega}(k)$ demand pairs and has size at most $\tau \cdot \tilde{O}(f(d, h))$, where $f(d, h) = \rho\tau/d + n/h + \rho h/d + \rho qd/\tau$.*

Proof. We know from Lemma 7 that the number of components of G' is at most $\tilde{O}(\rho\tau/d + n/h)$, and since each component of G' is contracted to a single terminal the total number of terminals in the Steiner Forest instance is also at most $\tilde{O}(\rho\tau/d + n/h)$. Thus by Lemma 8, we know that $|F'| \leq \tau \cdot \tilde{O}(\rho\tau/d + n/h)$. We also know from Lemma 7 that $|E'| \leq \tilde{O}(\rho h\tau/d + \rho qd)$ and that the algorithm covers $\tilde{\Omega}(k)$ demand pairs. This proves the theorem. ◀

The attentive reader will note that we only proved the ability to cover $\tilde{\Omega}(k)$ demand pairs, rather than k pairs. This is a minor technicality, though, as the next lemma shows.

▶ **Lemma 10.** *Suppose that* Steiner k-Forest *admits a bicriteria approximation algorithm that returns a subgraph of weight* $\leq f \cdot \tau$ *that connects at least* k/p *demand pairs, where* $1 < p < k$. *Then* Steiner k-Forest *admits an* $f \cdot \lceil \ln k / \ln \alpha \rceil$*-approximation algorithm, where* $\alpha = 1 + \frac{1}{p-1}$. *In particular, if* $k = n^{\epsilon}$ *for some* $\epsilon > 0$ *and* $p = \text{polylog}(n)$, *then* Steiner k-Forest *admits a* $\tilde{O}(f)$*-approximation algorithm.*

Proof. We run the bicriteria algorithm iteratively, as follows. Let k_i denote the residual demand (the number of pairs we still need to connect) at the beginning of iteration i, where $k_1 = k$. While $k_i \geq 1$, we run the bicriteria algorithm, remove from D the pairs connected in the current iteration, and set $k_{i+1} = k_i - p_i$, where p_i is the number of pairs connected at iteration i. Clearly, at the beginning of each iteration i there exists a solution to the residual problem (namely, a subgraph that connected k_i pairs from the remaining pairs) of weight at most τ, where τ is the optimal solution value to the original problem. Hence the weight of the bicriteria solution computed at each iteration is at most $f \cdot \tau$.

We have $k_i = k_{i-1} - p_{i-1} \leq k_{i-1} - (1 - 1/\alpha)k_{i-1} = k_{i-1}/\alpha$, hence $k_i \leq k/\alpha^i$. The least integer i such that $\alpha^i \geq k$ is $i = \lceil \ln k / \ln \alpha \rceil$, and it bounds the number of iterations. Consequently, the overall weight of the solution computed is at most $f \cdot \lceil \ln k / \ln \alpha \rceil \cdot \tau$, as claimed.

The last statement follows from the observation that for $p = \text{polylog}(n)$ we have $\ln \alpha = \ln(1 + \frac{1}{p-1}) \approx \frac{1}{p-1}$ and hence $\ln k / \ln \alpha \approx p \ln k = \text{polylog}(n)$. ◀

We now instantiate some parameters to show that for certain ranges of values, our algorithm gives a good approximation ratio.

▶ **Lemma 11.** *If* $\tau\sqrt{q} \leq n/\rho$ *then* Steiner k-Forest *with unit weights admits approximation ratio* $\tilde{O}\left(\left(\frac{\rho^2 nq}{\tau}\right)^{1/3}\right)$.

Proof. Let $f(d, h) = \rho\tau/d + n/h + \rho h/d + \rho q d/\tau$ be as in Theorem 9 and let

$$d = \left(\frac{n\tau^2}{\rho q^2}\right)^{1/3} \qquad \text{and} \qquad h = \left(\frac{n^2\tau}{\rho^2 q}\right)^{1/3} = d \cdot \left(\frac{nq}{\rho\tau}\right)^{1/3}.$$

Elementary computations show that

$$\frac{\rho\tau}{d} = \left(\frac{\rho^4 q^2 \tau}{n}\right)^{1/3} \qquad \text{and} \qquad \frac{n}{h} = \frac{\rho h}{d} = \frac{\rho q d}{\tau} = \left(\frac{\rho^2 nq}{\tau}\right)^{1/3}.$$

The statement follows from Theorem 9 and Lemma 10, since the condition $\tau\sqrt{q} \leq n/\rho$ implies $d, h \leq n/2$ and $\frac{\rho^4 q^2 \tau}{n} \leq \frac{\rho^2 nq}{\tau}$. ◀

▶ **Corollary 12.** *For any* $0 \leq \gamma \leq 1$, *the following holds: if* $\rho = n^{\gamma}$ *and* $\tau\sqrt{q} \leq n^{1-\gamma}$, *then* Steiner k-Forest *with unit weights admits approximation ratio* $\tilde{O}\left(n^{1/3+2\gamma/3}\right)$.

Proof. This follows from Lemma 11, since $\frac{\rho^2 nq}{\tau} = \frac{q}{\tau}\rho^2 n \leq 2\rho^2 n = 2n^{1+2\gamma}$. ◀

From Corollaries 12 and 6 it follows that Steiner k-Forest with unit weights admits approximation ratio $\tilde{O}\left(n^{1/3+2\gamma/3}\right)$, as claimed in Theorem 3. It only remains to prove Lemma 7.

4 Proof of Lemma 7

As discussed in Section 2, we will prove Lemma 7 by first constructing a clustering and then selecting a carefully chosen subset of that clustering. We begin by defining a clustering and proving a few simple results about them. We will then show how to use an algorithm for MCℓ-EPS to prove Lemma 7.

4.1 Clustering

▶ **Definition 13.** A (d, r)-*cluster* of a subset S of nodes in a graph G is a collection $\mathcal{T}_S$ of node-disjoint rooted subtrees of G of radius at most r each, such that $dist_G(u, v) > d$ for any two nodes $u, v \in S$ that belong to distinct trees. A (d, r)-*cluster-decomposition* of S is a collection of (d, r)-clusters $\{\mathcal{T}_A : A \in \mathcal{A}\}$ where $\mathcal{A}$ is a partition of S.

▶ **Lemma 14.** *There exists a polynomial time algorithm that given a graph $G = (V, E)$, a node subset $S \subseteq V$, and an integer $1 \leq d \leq n/2$ (called the* clustering parameter*), constructs a $(d, d(\lg |S| + 1))$-cluster $\mathcal{T}_A$ of a subset $A \subseteq S$ with $|A| \geq |S|/2$, where $\lg i = \log_2 i$.*

Proof. For a subtree T of G let $B_d(T) = \{v \in S \setminus T : \text{dist}_G(T, v) \leq d\}$ denote the set of nodes in $S \setminus T$ of distance at most d from T. The algorithm is as follows.

Algorithm 2: CLUSTER-CONSTRUCT(G, S, d)

```
initialize 𝒯 ← ∅, A ← ∅
while S ≠ ∅ do
    Choose root s ∈ S and set T ← ({s}, ∅)
    while |B_d(T)| ≥ |S ∩ T| do
        EXPAND(T): For each v ∈ B_d(T), add to T the shortest path from T to v.
UPDATE(𝒯, S, A): Add T to 𝒯, move T ∩ S from S to A, and remove B_d(T) from S.
return 𝒯
```

The lines in the loop add nodes to the trees as long as the number of terminals in the "boundary" is at least equal to the number of terminals inside the tree. When this is no longer the case, the update line removes the boundary of the new tree from the graph.

Each time we expand T, the radius of T increases by at most d while $|T \cap S|$ is at least doubled. Thus the radius of T is bounded by $d(\lg |S| + 1)$.

Note that at the update step, the set $B_d(T)$ of nodes within distance d from T is removed from S, and thus none of them will belong to A. This implies that at the end of the algorithm, $\text{dist}_G(u, v) > d$ for any two nodes $u, v \in A$ that belong to distinct trees. Note also that the number of nodes moved from S to A and included in T is at least half the number of nodes removed from S (since at this point $|B_d(T)| \leq |T \cap S|$). This implies that $|A| \geq |S|/2$.

It remains to prove that the trees in $\mathcal{T}$ are pairwise node-disjoint. Suppose to the contrary that there is $v \in V$ that belongs to two trees $T_1, T_2 \in \mathcal{T}$, where T_2 was constructed after T_1. Let T_2' denote the tree stored in T_2 right before the expansion step when v was added to T_2. When v was added to T_2', this was because there was a path of length $\leq d$ that goes through v from T_2' to some $t \in S$. In particular, $\text{dist}_G(v, t) \leq d$. Now let T_1' denote the tree stored in T_1 right after the expansion step when v was added to T_1. At this point, t was not added to T_1', hence we must have $\text{dist}_G(v, t) > d$. This is a contradiction. ◀

▶ **Corollary 15.** *There exists a polynomial time algorithm that given a graph $G = (V, E)$, a node subset $S \subseteq V$, and an integer $1 \leq d \leq n/2$, returns a $(d, d(\lg |S| + 1))$-cluster-decomposition of S with at most $\lg |S| + 1$ clusters.*

Proof. We construct the clusters in the decomposition sequentially, using the algorithm from Lemma 14. After construction of each cluster $\mathcal{T}_A$ we remove from S the corresponding set A of nodes and add A to $\mathcal{A}$. Clearly, at the end $\mathcal{A}$ is a partition of S. After each cluster construction the number of nodes in S decreases by a factor of at least 2, hence $|\mathcal{A}| \leq \lg |S| + 1$. ◀

Lemma 14 and Corollary 15 extend to edge-weighted graphs by an elementary construction of replacing every edge e of weight w_e by a path of length w_e.

▶ **Lemma 16.** *Given a* Steiner k-Forest *instance, let $\{\mathcal{T}_A : A \in \mathcal{A}\}$ be a cluster-decomposition of the set R of terminals as in Corollary 15 (so $|\mathcal{A}| \leq \lg |R| + 1$), and let D' be an arbitrary set of demand edges. Then there exist $A, B \in \mathcal{A}$ (possibly $A = B$) such that $\Omega(|D'|/\lg^2 |D|)$ pairs in D have one node in A and the other node in B.*

Proof. Let $D'(A, B)$ denote the set of pairs in D' with one node in A and the other in B. The statement follows by an averaging argument from the observations that $\sum_{\{A,B\} \subseteq \mathcal{A}} |D'(A, B)| = |D'|$, and that $|\{\{A, B\} : \{A, B\} \subseteq \mathcal{A}\}| = |\mathcal{A}|(|\mathcal{A}| + 1)/2$. ◀

For simplicity of exposition, let us assume that we know the sets A, B as in the above corollary (we can try all possible choices) and that $A \neq B$ (the analysis of the case $A = B$ is similar). Furthermore, by Lemma 16, we lose only a polylogarithmic factor by replacing D by $D(A, B)$; hence we assume that $D = D(A, B)$ (and that an optimal solution connects k pairs from D), and denote by $\mathcal{T}_A, \mathcal{T}_B$ the corresponding pair of clusters.

4.2 Choosing Trees

Now that we have two clusters which contain a lot of demand pairs, we want to find a cheap way of connecting much of it. Recall that J denotes an optimal solution, $\tau = |J|$ is the number of edges in J, D_J is a set of k demand pairs connected by J, R_J is the union of all pairs in D_J, and $q = |R_J|$. The two parameters d and h from Lemma 7 are related to the cluster decomposition, and have the following meaning:

- d is the cluster decomposition parameter as in Corollary 15.
- h is a threshold on tree size in a cluster; a tree T is *heavy* if it has more that h edges, and T is *light* otherwise.

Let us consider the case that $\text{dist}_J(u, v) \geq 2d$ holds for at least half of the pairs in D_J. In this case, we can remove from D all pairs $\{u, v\}$ with $\text{dist}_J(u, v) < 2d$, losing only a constant factor of 2 in the number of connected pairs. For simplicity of exposition, we will assume that $\text{dist}_J(u, v) \geq 2d$ holds for all pairs in D_J.

▶ **Lemma 17.** *Suppose that $\text{dist}_J(u, v) \geq 2d$ for every $\{u, v\} \in D_J$. Then at most τ/d trees in $\mathcal{T}_A$ contain a node from a pair in D_J.*

Proof. For every tree $T \in \mathcal{T}_A$ that intersects the optimum, fix some pair $\{u_T, v_T\} \in D_J$, where $u_T \in T$. Let P_T be the set of the first d nodes on the $u_T v_T$-path in J. Note that this path is completely unrelated to the trees in the cluster. Its a path in the optimum, and so we can not know what the path is. Nevertheless we get the following property: The sets P_T are disjoint, since the distance between any two terminals that belong to distinct trees is

larger than $2d$. Thus every tree that intersects J is associated with its own path of length d. Since the paths are edge disjoint and the number of total edges is at most τ, there are at most τ/d trees that intersect J. ◀

Let us partition D_J into three sets: D_{LL} of LL-pairs with both nodes in light trees, D_{HH} of HH-pairs with both nodes in heavy trees, and $D_{LH} = D_J \setminus (D_{LL} \cup D_{HH})$ of LH-pairs with one node in a light tree and the other in a heavy tree. At least one of these sets has size at least $k/3$. We execute three different algorithms, and choose the outcome of one of them. Intuitively, in each algorithm, we have the following three procedures.

1. Construct: This procedure constructs a MCℓ-EPS instance from the graph (R, D).
2. Compute: This procedure computes a ρ-approximate solution to the obtained MCℓ-EPS instance, which determines a certain set R' of terminals.
3. Connect: This procedure returns a graph $G' = (V', E')$ obtained by connecting some pairs of chosen terminals.

Algorithm 3: Heavy-Heavy$(G, D, \mathcal{T})$

1 Construct a MCℓ-EPS instance with $\ell = k/3$ by removing from the graph (R, D) nodes that belong to light trees.
2 Compute a ρ-approximate solution R' for the obtained MCℓ-EPS instance (in fact, here we get unit node-costs and unit edge-profits, so just a Mk-ES instance with $k = \ell$).
3 Connect: $G' = (V', E')$ is the union of the shortest paths from each terminal in R' to the root of the tree it belongs to.

Algorithm 4: Light-Light$(G, D, \mathcal{T})$

1 Construct a MCℓ-EPS instance with $\ell = k/3$ from the graph $H = (R, D)$ as follows.
- Remove nodes that belong to heavy trees.
- For every light tree, shrink its terminals into a single node.
- For every node pair u, v, replace the set D_{uv} of parallel uv-edges by a single edge of profit $|D_{uv}|$.

2 Compute a ρ-approximate solution R' for the obtained MCℓ-EPS instance (with unit node-costs and with edge profits).
3 Connect: $G' = (V', E')$ is the union of the light trees in $\mathcal{T}$ that correspond to R'.

Algorithm 5: Light-Heavy$(G, D, \mathcal{T})$

1 Construct a MCℓ-EPS instance with $\ell = k/3$ from the graph (R, D) as follows: for every light tree, shrink its terminals into a single node of cost dq/τ.
2 Compute a ρ-approximate solution R' for the obtained MCℓ-EPS instance (with node-costs in $\{1, dq/\tau\}$ and unit edge-profits).
3 Connect: $G' = (V', E')$ is the union of the light trees that correspond to nodes in R' and shortest paths from each terminal in R' that belongs to a heavy tree to the root of the heavy tree it belongs to.

▶ **Lemma 18.** *Suppose that* $\mathrm{dist}_J(u, v) \geq 2d$ *holds for every* $\{u, v\} \in D_J$.

(i) *If* $|D_{HH}| \geq k/3$ *then Algorithm 3 computes a graph* $G' = (V', E')$ *with* $O(n/h)$ *connected components,* $|E'| = \tilde{O}(\rho q d)$, *and* V' *contains* $\tilde{\Omega}(k)$ *pairs from* D.

(ii) *If* $|D_{LL}| \geq k/3$ *then Algorithm 4 computes a graph* $G' = (V', E')$ *with* $O(\rho\tau/d)$ *connected components,* $|E'| = O(\rho h\tau/d)$, *and* V' *contains* $\tilde{\Omega}(k)$ *pairs from* D.

(iii) *If* $|D_{LH}| \geq k/3$ *then Algorithm 5 computes a graph* $G' = (V', E')$ *with* $O(\rho\tau/d + n/h)$ *connected components,* $|E'| = \tilde{O}(\rho h\tau/d + \rho q d)$, *and* V' *contains* $\tilde{\Omega}(k)$ *pairs from* D.

Proof. Suppose that $|D_{HH}| \geq k/3$. The MCℓ-EPS instance in Algorithm 3 has a feasible solution of size at most q. Hence a ρ-approximate solution R' has size at most $|R'| \leq \rho q$. We connect each terminal in R' by a path of length $O(d \lg n)$, hence $|E'| = \tilde{O}(\rho q d)$. The heavy trees in each of $\mathcal{T}_A, \mathcal{T}_B$ are node disjoint, and each of them has at least h nodes. Thus G' has at most $2n/h$ connected components.

Suppose that $|D_{LL}| \geq k/3$. By Lemma 17, the MCℓ-EPS instance in Algorithm 4 has a feasible solution of size at most τ/d. Hence a ρ-approximate solution R' has size $|R'| \leq \rho\tau/d$, which bounds the number of trees from $\mathcal{T}$ and connected components that we include in G'. As the number of edges in each tree is at most h, $|E'| \leq \rho h\tau/d$.

Suppose that $|D_{LH}| \geq k/3$. Recall that in this case, in the obtained MCℓ-EPS instance, the cost of each node that corresponds to a light tree is $d \cdot q/\tau$, while the other nodes have unit costs and their number is at most q. By Lemma 17, at most τ/d trees contain a node from a pair in D_J. Hence the obtained MCℓ-EPS instance admits a solution of node cost $(\tau/d) \cdot d \cdot q/\tau = O(q)$. The returned ρ-approximate solution R' has node cost $O(\rho q)$, and thus $O(\rho q)$ nodes. The number of light trees obtained is therefore bounded by $\frac{O(\rho q)}{d \cdot q/\tau} = \frac{\tau \cdot \rho}{d}$, which coincides with our bound on the number light trees returned in the D_{LL} case. The number of connected components of heavy terminals is $O(\rho \cdot q)$ which coincides with our bound in the D_{HH} case. This proves the claim. ◀

This completes the analysis of the case that $\text{dist}_J(u, v) \geq 2d$ holds for at least half of the pairs in D_J.

Now let us consider the case when $\text{dist}_J(u, v) < 2d$ holds for at least half of the pairs in D_J. This is the case that most pairs can be connected by a relatively short path. In this case, we remove from D all pairs $\{u, v\}$ with $\text{dist}_J(u, v) \geq 2d$ in the graph. The algorithm in this case is essentially identical to Algorithm 3 as in case (i) of Lemma 18 (the $|D_{HH}| \geq k/3$ case). We construct a MCℓ-EPS instance as in Algorithm 3 with $m = k/2$. The obtained MCℓ-EPS instance admits a solution with q nodes, and the returned ρ-approximate solution R' has $O(\rho q)$ nodes. The same analysis as in case (i) of Lemma 18 finishes the proof of this case. Alternatively, we may also argue that since $\text{dist}_J(u, v) < 2d$ for all pairs in D, then by Lemma 8, we can connect $k/2$ pairs by cost $O(\rho q d)$, a term that already appears in Lemma 18(iii). Summarizing, the case when $\text{dist}_J(u, v) < 2d$ holds for at least half of the pairs in D_J appears in the analysis of Lemma 18(iii), and thus does not change the approximation ratio.

This ends the proof of Lemma 7, and thus proves Theorem 1.

5 Proof of Theorem 2

We first make the node costs bounded by a polynomial in n. We remove nodes of cost more than τ and zero the edges of cost at most τ/n^2. The cost we ignore due to the zeroing of the node costs is less than τ and is negligible in our context. Then we divide all the weights by the minimum weight and round the value down. Note that the cost of any edge over the cost of the minimum weight is at least 1. Hence the rounding down loses a negligible factor of 2: the worse case is that we may round a number that is at most 2 to 1. If the profits are exponential in n or larger, we give a bicriteria approximation in which we have the same ratio but we cover only $\ell - \ell/poly(n)$ profit where $poly(n)$ is an arbitrary polynomial function of n. Thus our generalization of [5] is really for the case when node weights are arbitrary and edge profits are polynomial in n.

We call an instance of MCℓ-EPS *simple* if all the edge-profits are the same and there are at most two distinct node costs (say c_1 and c_2) such that every edge has exactly one endpoint of each cost (note that it might be the case that $c_1 = c_2$).

▶ **Lemma 19.** *If* MCℓ-EPS *admits an f-approximation algorithm on simple instances, then* MCℓ-EPS *admits a bicriteria approximation algorithm that returns a graph of node-cost $O(f)$ times the optimal and edge-profit $\Omega(\ell/\log^3 n)$.*

Proof. Let $\langle G=(V,E), c, p, \ell\rangle$ be an instance of MCℓ-EPS. Recall that we may assume that the node costs are polynomial in n because of the reduction described above. Also, by assumption, the edge profits are bounded by a polynomial in n.

Partition E into $O(\log n)$ sets $E_h=\{e\in E: 2^h \le p_e < 2^{h+1}\}$. Each $e\in E_h$ is given profit 2^h. Partition the nodes similarly: $V_i=\{v\in E: 2^h \le c_v < 2^{h+1}\}$. according to powers of 2. Let E_{ijh} be the set of edges in E_h with one end in V_i and the other in V_j. The edge sets E_{ijh} partition E, and there are $O(\log^3 n)$ such sets. Each graph $G_{ijh}=(V_i\cup V_j, E_{ijh})$ gives a simple instance of MCℓ-EPS, and one of them contains $\Omega(\ell/\log^3 n)$ profit of the optimum. We run the algorithm for simple instances on each graph G_{ijh} with $\Omega(\ell/\log^3 n)$ instead of ℓ, and return the one of minimum node cost. The returned subgraph has node cost $O(f)$ times the optimal and $\Omega(\ell/\log^3 n)$ edge-profit, as required. ◀

By the same argument as in Lemma 10 we have the following.

▶ **Lemma 20.** *Suppose that* MCℓ-EPS *admits a bicriteria approximation algorithm that returns a graph of node-cost f times the optimal and edge-profit at least $(1-1/\alpha)\cdot\ell$, where $1<\alpha<\ell$. Then* MCℓ-EPS *admits an $f\cdot\lceil \ln \ell/\ln\alpha\rceil$-approximation algorithm.*

From Lemmas 19 and 20 we have the following.

▶ **Corollary 21.** *Suppose that* MCℓ-EPS *on simple instances admits a bicriteria approximation algorithm that returns a graph of node-cost f times the optimal and edge-profit $\tilde{\Omega}(\ell)$. Then* MCℓ-EPS *admits a $\tilde{O}(f)$-approximation algorithm.*

In the rest of this section we prove the following statement, which together with Corollary 21 implies Theorem 2.

▶ **Lemma 22.** MCℓ-EPS *on simple instances admits a bicriteria approximation algorithm that returns a graph of node-cost f times the optimal and edge-weight $\tilde{\Omega}(\ell)$, where $f=\tilde{O}\left(n^{3-2\sqrt{2}+\epsilon}\right)$ for arbitrarily small constant $\epsilon>0$.*

We need some definitions and results from [5].

▶ **Definition 23** ([5])**.** A bipartite graph $G=(V_1\cup V_2, E)$ is called (n_1,d_1,n_2,d_2)-*nearly regular* if for every $i=1,2$ we have $|V_i|=n_i$ and the following condition on the degrees holds:

$$d_i \ge \max_{v\in V_i} d(v) \ge \min_{v\in V_i} d(v) = \Omega(d_i/\log n).$$

▶ **Lemma 24** ([5])**.** *Any graph $H=(V,E)$ contains an (n_1,d_1,n_2,d_2)-nearly regular subgraph with $\Omega(|E|/\log^2 n)$ edges, for some n_1,d_1,n_2,d_2.*

A key step in [5] was the following lemma:

▶ **Lemma 25** ([5])**.** *For any $\epsilon>0$ there exists a randomized polynomial time algorithm that given a bipartite graph G on n nodes that contains an (n_1,d_1,n_2,d_2)-nearly regular subgraph, returns a subgraph $G'=(V',E')$ of G such that $|V'|\le f\cdot(n_1+n_2)$ (with probability 1) and $\mathbf{E}[|E'|]=\tilde{\Omega}(n_1d_1)$, where $f=n^{3-2\sqrt{2}+\epsilon}$.*

We prove the following refinement of Lemma 25, which gives a more "balanced" guarantee.

▶ **Lemma 26.** *For any $\epsilon > 0$ there exists a randomized polynomial time algorithm that given a bipartite graph G on n nodes that contains an (n_1, d_1, n_2, d_2)-nearly regular subgraph, returns a subgraph $G' = (V', E')$ of G such that $|V' \cap V_1| \leq f n_1$ and $|V' \cap V_2| \leq f n_2$ (with high probability) and $\mathbf{E}[|E'|] = \tilde{\Omega}(n_1 d_1)$, where $f = n^{3-2\sqrt{2}+\epsilon}$.*

Proof. We will assume that $n_1 \geq n_2$; otherwise we just switch indices. Note that if $n_1 \leq 2n_2$, then the algorithm from Lemma 25 produces a subgraph that satisfies the new stronger requirement on the chosen nodes. So suppose that $n_1 > 2n_2$. For simplicity, let us also assume that $p = n_1/n_2$ is an integer.

Let $\hat{G} = (V_1 \cup \hat{V}_2, \hat{E})$, where $\hat{V}_2$ consists of p copies of V_2 and $\hat{E}$ is obtained by putting between V_1 and each copy of V_2 a copy of E. For a subgraph $G' = (V_1' \cup V_2', E')$ of G let $\hat{G}' = (V_1' \cup \hat{V}_2', \hat{E}')$ denote the corresponding subgraph of $\hat{G}$, i.e. where between each copy of V_1' and V_2' we include a copy of E'. Note that $|\hat{V}_2'| = p|V_2'|$, that $d_{\hat{G}'}(v) = p d_{G'}(v)$ if $v \in V_1'$, and that if $\hat{v} \in \hat{V}_2$ is a copy of $v \in V_2$ then $d_{\hat{G}'}(\hat{v}) = d_{G'}(v)$. This implies that if G' is (n_1, d_1, n_2, d_2)-nearly regular then $\hat{G}'$ is (n_1, d_2, n_1, d_2)-nearly regular.

We run the algorithm from Lemma 25 on the instance $\langle \hat{G}, (n_1, d_2, n_1, d_2)\rangle$ independently $\tilde{O}(n^2)$ times, and among the subgraphs computed take one $\hat{G}' = (V_1' \cup \hat{V}_2', \hat{E}')$ with maximum number of edges. For each $v \in V_2$, let T_v denote the number of copies of v in $\hat{V}_2'$. We build V_2' by sampling each $v \in \hat{V}_2'$ independently with probability T_v/p. Let E' be the set of edges between V_1' and V_2'. We will return the graph $G' = (V_1' \cup V_2', E')$.

We now prove the bounds on the sizes of V_1', V_2', and E'. Since we run the algorithm as in Lemma 25, $|V_1'| \leq 2f n_1$ and $\sum_{v \in V_2} T_v \leq |\hat{V}_2'| \leq 2f n_1$. By linearity of expectations, we get that the expected size of V_2' is at most $\frac{n_2}{n_1} \sum_{v \in V_2} T_v \leq 2f n_2$. Since each node in V_2 was chosen independently, a simple Chernoff bound implies that $|V_2'| = \tilde{O}(f n_2)$ with high probability.

To bound $|E'|$, note that a Chernoff bound implies that with high probability $|\hat{E}'| = \tilde{\Omega}(n_1 d_2)$ (since we ran Lemma 25 a polynomial number of times and took the best, and each run was independent). An edge $uv \in E$ with $u \in V_1'$ is included in our subgraph with probability $T_v/p = T_v n_2/n_1$. Thus

$$\begin{aligned}\mathbf{E}[|E'|] &= \sum_{u \in V_1'} \sum_{v \in V_1' : uv \in E} T_v/p = \frac{n_2}{n_1} \sum_{u \in V_1'} \sum_{v \in V_2' : uv \in E} T_v \\ &= \frac{n_2}{n_1} \cdot \tilde{\Omega}(n_1 d_2) = \tilde{\Omega}(n_2 d_2) = \tilde{\Omega}(n_1 d_1),\end{aligned}$$

proving the lemma. ◀

Now we finish the proof of Lemma 22. Let $\langle G = (V_1 \cup V_2, E), m, (c_1, c_2)\rangle$ be a simple MCℓ-EPS instance. Let $G^* = (V_0^* \cup V_1^*, E^*)$ be an optimal subgraph. Applying Lemma 24 to G^* implies that there exist values of n_1, d_1, n_2, d_2 such that there is a (n_1, d_1, n_2, d_2)-nearly regular subgraph of G of cost at most $c(V^*) = c_1|V_1^*| + c_2|V_2^*|$ that contains at least $\tilde{\Omega}(\ell) = \tilde{\Omega}|E^*|$ edges (note that up to polylogs $n_1 d_1 = n_2 d_2 = \ell = |E^*|$). So when we run the algorithm from Lemma 26, we get a graph $G' = (V_1' \cup V_2', E')$ with the properties that with high probability $|V_1'| = \tilde{O}(f n_1)$ and $|V_2'| = \tilde{O}(f n_2)$ and in expectation $|E'| = \tilde{\Omega}(n_1 d_1) = \tilde{\Omega}(\ell)$. The node-cost of this subgraph is $\tilde{O}(f n_1 c_1 + f n_2 c_2) = \tilde{O}(f) \cdot c(V^*)$. This proves Lemma 22, and thus also the proof of Theorem 2 is complete.

References

1 A. Agrawal, P. Klein, and R. Ravi. When trees collide: an approximation algorithm for the generalized Steiner problem on networks. *SIAM J. Computing*, 24(3):440–456, 1995.

2 I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993.

3 Jaroslaw Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità. An improved LP-based approximation for Steiner tree. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, STOC'10, pages 583–592, 2010.

4 M. Charikar and B. Raghavachari. The finite capacity dial-a-ride problem. In *FOCS*, pages 458–467, 1998.

5 E. Chlamtac, M. Dinitz, and R. Krauthgamer. Everywhere-sparse spanners via dense subgraphs. In *FOCS*, pages 758–767, 2012.

6 M. Feldman, G. Kortsarz, and Z. Nutov. Improved approximation algorithms for directed Steiner forest. *J. Comput. Syst. Sci.*, 78(1):279–292, 2012.

7 N. Garg. Saving an epsilon: a 2-approximation for the k-MST problem in graphs. In *STOC*, pages 396–402, 2005.

8 A. Gupta, M. T. Hajiaghayi, V. Nagarajan, and R. Ravi. Dial a ride from k-forest. *ACM Transactions on Algorithms*, 6(2), 2010.

9 M. T. Hajiaghayi and K. Jain. The prize-collecting generalized Steiner tree problem via a new approach of primal-dual schema. In *SODA*, pages 631–640, 2006.

Computing Opaque Interior Barriers à la Shermer

Adrian Dumitrescu*[1], Minghui Jiang[2], and Csaba D. Tóth[3]

1 **Department of Computer Science, University of Wisconsin–Milwaukee, U.S.A**
dumitres@uwm.edu
2 **Department of Computer Science, Utah State University, Logan, U.S.A**
mjiang@cc.usu.edu
3 **Department of Mathematics, California State University, Northridge, CA; and Department of Computer Science, Tufts University, Medford, MA, U.S.A**
cdtoth@acm.org

Abstract

The problem of finding a collection of curves of minimum total length that meet all the lines intersecting a given polygon was initiated by Mazurkiewicz in 1916. Such a collection forms an *opaque barrier* for the polygon. In 1991 Shermer proposed an exponential-time algorithm that computes an interior-restricted barrier made of segments for any given convex n-gon. He conjectured that the barrier found by his algorithm is optimal, however this was refuted recently by Provan et al. Here we give a Shermer like algorithm that computes an interior polygonal barrier whose length is at most 1.7168 times the optimal and that runs in $O(n)$ time. As a byproduct, we also deduce upper and lower bounds on the approximation ratio of Shermer's algorithm.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Opaque barrier, approximation algorithm, isoperimetric inequality

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.128

1 Introduction

The problem of finding small sets that block every line passing through a unit square was first considered by Mazurkiewicz in 1916 [21]; see also [2, 14]. Let B be a convex body in the plane. Following Bagemihl [2], a set Γ is an *opaque set* or a *barrier* for B, if Γ meets every line that intersects B. We restrict our attention to barriers consisting of countably many rectifiable curves. A barrier does not need to be connected; it may consist of one or more rectifiable arcs and its parts may lie anywhere in the plane, including the exterior of B [2].

What is the length of the shortest barrier for a given convex body B? In spite of considerable efforts, the answer to this question is not known even in the simplest instances, such as when B is a square, a disk, or an equilateral triangle; see [3], [4, Problem A30], [10], [11], [12], [13, Section 8.11], [15, Problem 12]. For example, when B is a unit square, the barrier in Figure 1(right) is conjectured to be optimal; on the other hand, the current best lower bound on the length of a barrier was only 2 until very recently; the earliest record for this bound of 2 dates back to Jones in 1964 [16]. For barriers consisting of finitely many straight-line segments and restricted to lie in a concentric square of side 2, Dumitrescu and Jiang [8] established the first lower bound greater than 2, namely $2 + 10^{-12}$ (and $2 + 10^{-5}$ for interior barriers). Kawamura et al. [17] recently obtained the first general lower bound (that does not require finiteness or locality), namely 2.00002, which now holds the current record.

* Supported in part by NSF grant DMS-1001667.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 128–143

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

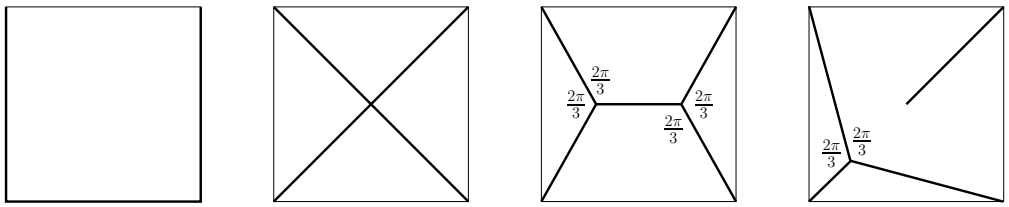


Figure 1 The first three from the left are barriers for the unit square of lengths 3, $2\sqrt{2} = 2.8284\ldots$, and $1+\sqrt{3} = 2.7320\ldots$. Right: The diagonal segment $[(1/2, 1/2), (1, 1)]$ together with three segments connecting the corners $(0, 1)$, $(0, 0)$, $(1, 0)$ to the point $(\frac{1}{2} - \frac{\sqrt{3}}{6}, \frac{1}{2} - \frac{\sqrt{3}}{6})$ yield a barrier of length $\sqrt{2} + \frac{\sqrt{6}}{2} = 2.6389\ldots$.

A barrier blocks any line of sight across the region B or detects any ray that passes through it. Potential applications are in guarding and surveillance [5]. Some applications to the optimization of saving and recovery routes can be found in [19, 20]. The problem of short barriers has attracted researchers for decades. We refer the reader to [7, 9] and the references therein for many of the earlier results in this area.

Types of Barriers. Several variants of the problem have been considered depending on the types of curves allowed in a barrier: the most restricted are the barriers made of single continuous arcs, then connected barriers, and lastly, arbitrary (possibly disconnected) barriers. For the unit square, the shortest known barrier in these three categories have lengths 3, $1 + \sqrt{3} = 2.7320\ldots$ and $\sqrt{2} + \frac{\sqrt{6}}{2} = 2.6389\ldots$, respectively. They are depicted in Figure 1. Obviously, disconnected barriers offer the greatest freedom of design. For instance, Kawohl [18] showed that the barrier in Figure 1(right) is optimal in the class of curves with at most two components restricted to lie in the square. For the unit disk, the shortest known barrier consists of three arcs. See also [11, 13].

Barriers can be also classified by their possible locations. In certain instances, it might be infeasible to construct barriers guarding a specific domain outside the domain, since that part might be controlled by different owners. An *interior barrier* of a body B is a barrier constrained to the interior and the boundary of B. For example, all four barriers for the unit square illustrated in Figure 1 are interior barriers. By slightly relaxing the interior constraint, we call a barrier for B, $(1+\varepsilon)$-*interior*, if it lies in the interior or on the boundary of $B + D_\varepsilon$, the Minkowski sum of B and a disk D_ε of radius $\varepsilon > 0$ centered at the origin.

On the other hand, certain instances may prohibit barriers lying in the interior of a domain. An *exterior barrier* of B is constrained to exterior and the boundary of B. As an illustration, the first barrier from the left in Figure 1 is exterior (and since it is contained in the boundary of the domain, it is interior as well).

Approximations. In the absence of methods for finding optimal barriers, attention has turned to approximation algorithms. A key fact in establishing a constant approximation ratio is the following lower bound on the length of a barrier: Every barrier Γ of a convex body B in the plane satisfies

$$|\Gamma| \geq \frac{\mathrm{per}(B)}{2}, \tag{1}$$

where $\mathrm{per}(B)$ denotes the perimeter of B. The proof of (1) is folklore, based on Cauchy's integral formula [9, 12, 16]. It follows that the boundary of B, of length $\mathrm{per}(B)$, is always

a 2-approximation to the optimal barrier, and a 2-approximation to the optimal *interior* barrier as well.

Recent work focused on obtaining better approximation ratios. Even though we have so little control on the shape or length of optimal barriers, barriers whose lengths are relatively close to optimal can be computed efficiently for a convex polygon P with n vertices. Various approximation algorithms with a smaller constant ratio (below 1.6) have been obtained recently [9]:

- **(i)** A (possibly disconnected) barrier for P, whose length is at most $\frac{1}{2} + \frac{2+\sqrt{2}}{\pi} = 1.5867\ldots$ times the optimal, can be computed in $O(n)$ time.
- **(ii)** A connected polygonal barrier whose length is at most 1.5716 times the optimal can be computed in $O(n)$ time.
- **(iii)** A single-arc polygonal barrier whose length is at most $\frac{\pi+5}{\pi+2} = 1.5834\ldots$ times the optimal can be computed in $O(n)$ time.
- **(iv)** An optimal interior single-arc barrier can be computed in $O(n^2)$ time.
- **(v)** An interior connected barrier whose length is at most $(1+\varepsilon)$ times the optimal can be found in polynomial time.

It is worth noting that none of the above approximations holds for interior barriers, for two reasons: either (i) the barrier found is not guaranteed to be interior, or (ii) the approximation ratio is based on a lower bound for a specific class of barriers, different from that given by (1), or both. In this paper, we present the first nontrivial approximation algorithm with ratio below 2 for computing an interior barrier for a given convex polygon.

▶ **Theorem 1.** *Given a convex polygon P with n vertices, an interior barrier for P, whose length is at most $0.8588\,\mathrm{per}(P) = 1.7168\,\frac{\mathrm{per}(P)}{2}$, hence in particular at most 1.7168 times the optimal, can be computed in $O(n)$ time.*

Shermer's Conjecture. In the late 1980s, Akman [1] soon followed by Dublish [6] had reported algorithms for computing a minimum interior barrier of a given convex polygon. Both algorithms were shown to be suboptimal by Shermer [25] in 1991, who proposed a new exact algorithm. Shermer conjectured that a shortest interior barrier (he calls it an "opaque forest") of a given convex polygon P with n vertices can be generated by an instance of the following procedure:

- **(a)** Find a triangulation T of P.
- **(b)** Remove zero or more diagonals of T, so that at most one nontriangular interior region U is formed. Let the edges of U's Steiner tree be in the opaque forest.
- **(c)** For all triangles of T (other than U, if U is a triangle), let the height of the triangle (using the edge topologically closest to U as the base) be in the opaque forest.

Equivalently, the algorithm proposed by Shermer is the following: For all possible subsets of 3 or more vertices of P, compute the convex hull $U \subset P$. For each such U, include the minimum Steiner tree of U in the barrier Γ and triangulate $P \setminus U$ in all possible ways. For each fixed triangulation, include the height of each triangle (using the edge topologically closest to U as the base) in Γ. Return the shortest interior barrier obtained in this way. (Note that not all choices of U and the triangulation of $P \setminus U$ produce an interior barrier. For example, if the triangulation contains an obtuse triangle with the base incident to the obtuse angle, then the corresponding height is in the exterior of the triangle and the polygon. However, the Shermer's method always produces an interior barrier for $U = P$.)

Recently, Provan et al. [23] refuted Shermer's conjecture with a convex polygon as simple as a rhombus. Specifically, their example shows that Shermer's procedure does not always

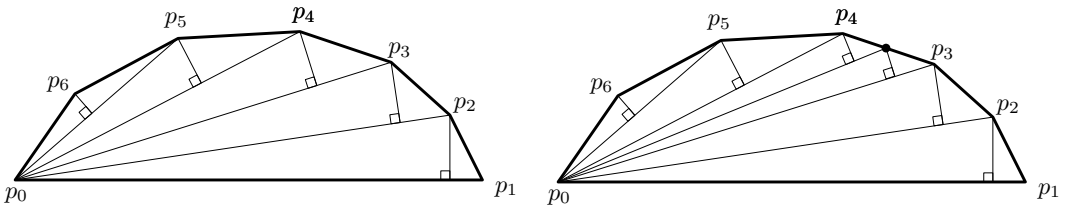


Figure 2 Left: A fan triangulation and the corresponding heights. Right: Edge p_3p_4 is subdivided by a dummy vertex.

compute the shortest interior barrier or the shortest unrestricted barrier. In Section 4, we slightly refine the lower bound for the approximation ratio of Shermer's algorithm offered by the example of Provan et al. Moreover, it is easy to replicate this lower bound with polygons with any number of vertices $n \geq 4$.

▶ **Theorem 2.** *There exist convex polygons (e.g., a rhombus) for which Shermer's algorithm returns an interior barrier that is at least* 1.00769 *times longer than the optimal.*

It is well-known that the number of triangulations of a convex n-gon is C_{n-2}, where $C_n = \frac{1}{n+1}\binom{2n}{n}$ is the the nth Catalan number, hence the proposed algorithm runs in time $O^*(4^n)$ (the $O^*()$ notation hides polynomial factors). However, it is easy to avoid the exponential running time while still achieving an approximation ratio well below 2. The approximation algorithm we will present constructs a barrier by a procedure similar to Shermer's, where some steps are relaxed. Moreover, the approximation ratio we derive also holds for Shermer's original algorithm.

▶ **Corollary 3.** *The approximation ratio of Shermer's procedure is at most 1.7168 and at least 1.00769.*

2 Preliminaries

Our main tool is an upper bound on the sum of heights produced by Shermer's procedure in a specific triangulation of a convex polygon.

Let $P = (p_0, p_1, \ldots, p_m)$ be a convex polygon where the vertices are labeled in counterclockwise order. The *fan triangulation* of P is obtained by inserting the chords p_0p_i, $i = 2, \ldots, m-1$, as shown in Figure 2.

Let h_i denote the distance from p_i to the supporting line of p_0p_{i-1} for $i = 2, \ldots, m$. The shortest segments between p_i and line p_0p_{i-1} are the *heights* corresponding to the fan triangulation. We first give a sufficient condition for the heights to lie in the interior of P.

▶ **Lemma 4.** *Assume that $P = (p_0, p_1, \ldots, p_m)$ lies in a half-disk of diameter p_0p_1. Then every triangle (p_0, p_{i-1}, p_i), $i = 2, \ldots, m$, has a right or obtuse angle at p_i; consequently the heights of the fan triangulation of P lie in the interior or on the boundary of P.*

Proof. By Thales' theorem, we have $\angle p_0p_ip_1 \geq \pi/2$ for $i = 2, \ldots, m$. Hence $\angle p_0p_ip_{i-1} \geq \angle p_0p_ip_1 \geq \pi/2$, and every triangle (p_0, p_{i-1}, p_i) has an obtuse or right angle at p_i, as required. ◀

▶ **Lemma 5.** *Assume that $P = (p_0, p_1, \ldots, p_m)$ lies in a half-disk of diameter p_0p_1. Let $\alpha = \angle p_1p_0p_m < \pi/2$ and $A = \text{area}(P)$. Then $\sum_{i=2}^{m} h_i \leq \sqrt{2\alpha A}$.*

Proof. Put $a_i = |p_0 p_i|$, $i = 1, \ldots, m$, and $\alpha_i = \angle p_{i-1} p_0 p_i$, $i = 2, \ldots, m$. Refer to Figure 2. Observe that $\sum_{i=2}^{m} \alpha_i = \alpha$. We have

$$\sum_{i=2}^{m} h_i = \sum_{i=2}^{m} a_i \sin \alpha_i. \tag{2}$$

Since P is subdivided into (the fan of) $m-1$ triangles with common vertex p_0, we also have

$$\sum_{i=2}^{m} a_{i-1} a_i \sin \alpha_i = 2A. \tag{3}$$

Since P lies in a half-disk of diameter $p_0 p_1$, we have $a_{i-1} < a_i$, for $i = 2, \ldots, m$. Consequently

$$\sum_{i=2}^{m} a_i^2 \sin \alpha_i \leq 2A. \tag{4}$$

By the Cauchy-Schwarz inequality in the first step and by Jensen's inequality for the sin function in the second step, (2) and (4) imply that

$$\begin{aligned} \left(\sum_{i=2}^{m} h_i\right)^2 = \left(\sum_{i=2}^{m} a_i \sin \alpha_i\right)^2 &\leq \left(\sum_{i=2}^{m} a_i^2 \sin \alpha_i\right) \left(\sum_{i=2}^{m} \sin \alpha_i\right) \\ &\leq (2A) \left((m-1) \sin \frac{\alpha}{m-1}\right) \leq (2A) \left((m-1) \frac{\alpha}{m-1}\right) = 2\alpha A. \end{aligned} \tag{5}$$

The required inequality follows by taking square roots. ◀

We present an alternative proof for Lemma 5, via an integral formula, which gives a tighter bound when the points p_i lie on an integrable curve.

▶ **Lemma 6.** *Assume that $P = (p_0, p_1, \ldots, p_m)$ lies in a half-disk of diameter $p_0 p_1$, such that p_0 is at the origin, p_1 is on the positive x-axis. Let $\alpha = \angle p_1 p_0 p_m < \pi/2$ and $A = \text{area}(P)$. Parametrize the polygonal arc $(p_1, \ldots, p_m)$ in polar coordinate by $(\theta, \lambda(\theta))$ for $\theta \in [0, \alpha]$. Then $\sum_{i=2}^{m} h_i \leq \int_0^{\alpha} \lambda(\theta)\, d\theta \leq \sqrt{2\alpha A}$.*

Proof. Put $\lambda_i = |p_1 p_i|$, $i = 2, \ldots, m$, and $\theta_i = \angle p_1 p_0 p_i$, $i = 2, \ldots, m$. Then we have

$$\sum_{i=2}^{m} h_i = \sum_{i=2}^{m} \lambda_i \sin(\theta_i - \theta_{i-1}) \leq \sum_{i=2}^{m} \lambda_i (\theta_i - \theta_{i-1}). \tag{6}$$

By Lemma 4, every triangle (p_0, p_{i-1}, p_i), $i = 2, \ldots, m$, has a right or obtuse angle at p_i. If we successively subdivide an edge $p_i p_{i+1}$, $i = 1, \ldots, m-1$, with dummy vertices (see Figure 2), then the sum of heights increases. By an infinite refinement of the polygonal arc $(p_1, \ldots, p_m)$ with dummy vertices, we obtain

$$\sum_{i=2}^{m} \lambda_i (\theta_i - \theta_{i-1}) \leq \int_0^{\alpha} \lambda(\theta)\, \mathrm{d}\theta. \tag{7}$$

By the Cauchy-Schwarz inequality,

$$\int_0^{\alpha} \lambda(\theta)\, \mathrm{d}\theta \leq \left(\int_0^{\alpha} \mathrm{d}\theta\right)^{1/2} \left(\int_0^{\alpha} \lambda^2(\theta)\, \mathrm{d}\theta\right)^{1/2} = \sqrt{\alpha} \left(\int_0^{\alpha} \lambda^2(\theta)\, \mathrm{d}\theta\right)^{1/2}. \tag{8}$$

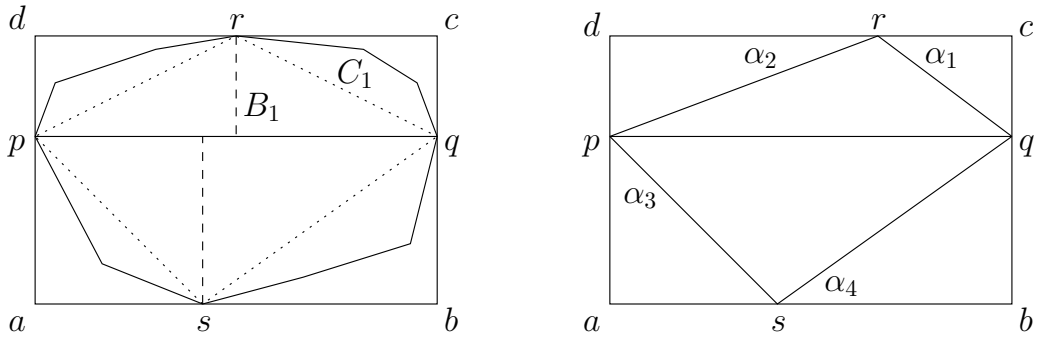


Figure 3 Left: eight areas in a rectangle. Right: four angles in a rectangle.

We have

$$\sum_{i=2}^{m} h_i \le \int_0^{\alpha} \lambda(\beta)\, d\beta, \quad 2A = \int_0^{\alpha} \lambda^2(\beta)\, d\beta, \quad \alpha = \int_0^{\alpha} d\beta, \tag{9}$$

and consequently, (8) yields $\sum_{i=2}^{m} h_i \le \sqrt{2\alpha A}$. ◀

3 Proof of Theorem 1

By the classic isoperimetric inequality [26, Exercises 5–8, 7–17] (or see [24]), we have

$$A \le \frac{L^2}{4\pi}, \text{ and } L \le \pi D, \tag{10}$$

where $L = \mathrm{per}(P)$, $A = \mathrm{area}(P)$, and $D = \mathrm{diam}(P)$.

We can assume without loss of generality that P has a horizontal unit diameter pq, and let $Q = abcd$ be a minimal axis-parallel rectangle (of unit width) containing P. Denote the height of Q by $y \le 1$; refer to Figure 3.

Let r and s be two points of P on the top and bottom sides of Q, respectively. Let α_i, $i = 1, 2, 3, 4$, denote the smallest acute angles in each of the four right triangles incident to the vertices of Q: Δqcr, Δrdp, Δpas, Δsbq. If B_i, C_i, $i = 1, 2, 3, 4$, are the areas indicated in the figure, write

$$A_i = B_i + C_i, \quad i = 1, \ldots, 4, \text{ and } B = \sum_{i=1}^{4} B_i, \quad C = \sum_{i=1}^{4} C_i, \text{ so that } A = \sum_{i=1}^{4} A_i.$$

We have $\mathrm{area}(P) = \sum_{i=1}^{4} A_i = A = B + C$. Observe that $C_i \le B_i$, for each $i = 1, 2, 3, 4$, hence $2C_i \le B_i + C_i = A_i$, for each $i = 1, 2, 3, 4$. Consequently, $2C \le A$.

Approximation Algorithm. Given a convex polygon P with n vertices, consider four interior barriers Γ_i, $i = 1, 2, 3, 4$, defined as follows: Γ_i consists of the boundary of P in three "quarters" of Q and the heights of a fan triangulation in the fourth quarter from the smallest angle made at the two endpoints of the convex chain. The algorithm outputs the shortest one.

Note that Γ_i, $i = 1, 2, 3, 4$, is a valid interior barrier for P: the boundary of P in three quarters of Q (say, all quarters but the ith) blocks any line intersecting $\mathrm{conv}(P) \setminus C_i$; and the fan of heights in C_i is a barrier by Shermer's argument [25] (this can be shown by an easy

inductive argument). By Lemma 4, all heights in the fan triangulations lie in the interior or on the boundary of P, so Γ_i, $i = 1, 2, 3, 4$, is an interior barrier. For a given polygon P with n vertices, the four barriers can be computed in $O(n)$ time: indeed, a diameter pair, an axis-aligned bounding box, the fan triangulations, and the heights can all be computed in $O(n)$ time.

First Bound on the Approximation Ratio. In the i-th quarter of Q, the smallest angle made at the two endpoints of the convex chain is at most α_i, for $i = 1, 2, 3, 4$. By Lemma 5, the sum of heights in the fan triangulation in the ith quarter is at most $\sqrt{2\alpha_i C_i}$. The length of the interior barrier Γ returned by the algorithm is no more than the average length of the four candidates:

$$4|\Gamma| \leq \sum_{i=1}^{4} |\Gamma_i| \leq 3L + \sum_{i=1}^{4} \sqrt{2\alpha_i C_i}. \tag{11}$$

By the Cauchy-Schwarz inequality we have

$$\sum_{i=1}^{4} \sqrt{2\alpha_i C_i} \leq \sqrt{\left(\sum_{i=1}^{4} \alpha_i\right)\left(2\sum_{i=1}^{4} C_i\right)} = \sqrt{\Lambda \cdot 2C}, \tag{12}$$

where $\Lambda := \sum_{i=1}^{4} \alpha_i$. Since $\alpha_i \leq \pi/4$, for each $i = 1, 2, 3, 4$, we have $\Lambda = \sum_{i=1}^{4} \alpha_i \leq \pi$. Recall that $2C \leq A$. These two bounds together with the first inequality in (10) yield the following upper bound on the right hand side of (12):

$$\sqrt{\Lambda \cdot 2C} \leq \sqrt{\pi A} \leq \frac{1}{2}L. \tag{13}$$

Hence by using (11), (12), and (13), it follows that the approximation ratio ρ is at most

$$\frac{|\Gamma|}{L/2} \leq \frac{3L + \sqrt{\Lambda \cdot 2C}}{2L} \leq \frac{3L + 0.5L}{2L} = 1.75. \tag{14}$$

A Refined Bound on the Approximation Ratio. We derive sharper bounds on both factors Λ and $2C$ that appear in (12).

The first key component is establishing an upper bound on the sum of angles $\Lambda = \sum_{i=1}^{4} \alpha_i$.

▶ **Lemma 7.** *Define the function*

$$f(y) = \begin{cases} \dfrac{1}{2} + \dfrac{\arctan \frac{y}{1-y}}{\pi}, & y \in [0, 1/2], \\ \dfrac{3}{4} + \dfrac{\arctan \frac{2y-1}{3-2y}}{\pi}, & y \in [1/2, 1]. \end{cases}$$

Then $\sum_{i=1}^{4} \alpha_i \leq f(y)\,\pi$. This inequality is the best possible for all $y \in (0, 1]$.

Proof. Observe that $f(1/2) = 3/4$, and that $f(y)$ is continuous at $y = 1/2$. Denote by y_1, y_2 the vertical distances from pq to the top and bottom side of Q, respectively. We can assume without loss of generality that $y_1 \leq y_2$. Put $x_1 = |cr|$, $x_2 = |rd|$, $x_3 = |as|$, $x_4 = |sb|$, so that $x_1 + x_2 = x_3 + x_4 = 1$. Refer to Figure 4. Denote by ℓ the supporting line of pq. We can assume without loss of generality that $|dr| \leq |rc|$ and $|as| \leq |sb|$ (by applying a reflection above and below the line ℓ, independently). This implies that $\alpha_1 = \angle crq$. We distinguish two cases depending on whether y_2 is larger or smaller than $1/2$.

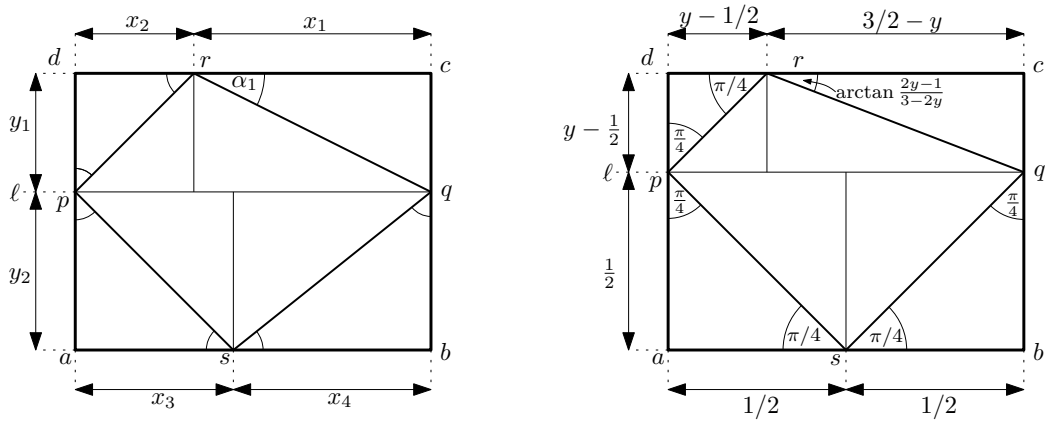


Figure 4 Left: the setup used in the proof of Lemma 7. Right: the canonical configuration in Case 1 where $y_2 \geq 1/2$.

Case 1: $y_2 \geq 1/2$. We first show that $\Lambda = \sum_{i=1}^{4} \alpha_i = \pi/2 + \arctan\frac{2y-1}{3-2y}$ can be attained. Consider the configuration in Figure 4, where $y_1 = x_2 = y - 1/2$, $y_2 = x_3 - x_4 = 1/2$, and $x_1 = 3/2 - y$. We call this the *canonical* configuration in Case 1. In the canonical configuration, Δdpr, Δasp, and Δbqs are isosceles right triangles, hence $\alpha_2 = \alpha_3 = \alpha_4 = \pi/4$, and $\alpha_1 = \arctan\frac{2y-1}{3-2y}$.

It is enough to show that the canonical configuration maximizes Λ under the constraint that $y_2 \geq 1/2$. Consider an arbitrary configuration with $y_2 \geq 1/2$. Keeping the bounding box Q fixed, move the points p, q, r, and s continuously to the canonical configuration such that $\alpha_1 + \alpha_2$ monotonically increases, and $\alpha_3 + \alpha_4$ increases overall during the transformation. Note that α_i, $i = 1, 2, 3, 4$, does not necessarily correspond to the same angle during a continuous transformation: a change can occur when $\alpha_i = \pi/4$. However, the value of α_i does change continuously.

Note that $\alpha_3 + \alpha_4 \leq \pi/4 + \pi/4 = \pi/2$. Therefore $\alpha_3 + \alpha_4$ is nondecreasing over all, no matter how we move p, q, r, and s to the canonical configuration. For changes in $\alpha_1 + \alpha_2$, we distinguish two subcases.

Case 1.1: $\alpha_2 = \angle dpr \leq \pi/4$. Move r to the right until $\angle dpr = \angle drp = \pi/4$. Observe that both α_1 and α_2 monotonically increase, hence $\alpha_1 + \alpha_2$ also monotonically increases. Next, move p, q, and r simultaneously such that pq remains horizontal and (d, p, r) remains an isosceles right triangle until p (hence p and q) reaches their canonical position. Observe that $\alpha_1 + \alpha_2$ monotonically increases. Finally, move s to its canonical position (which affects neither α_1 not α_2).

Case 1.2: $\alpha_2 = \angle drp \leq \pi/4$. This means that p and q are above their canonical position. Move p and q down simultaneously such that pq remains horizontal, until either they reach their canonical position or $\angle drp = \pi/4$. In the latter alternative, the proof can be finalized as in Case 1.1. If p and q are at their canonical position but $\angle drp < \pi/4$, then $x_2 \leq 1/2 \leq x_1$, and $\alpha_2 = \angle drp$. Move r to the left until r reaches its canonical position. Consider the circular arc determined by the three points p, r, q. Since r remains on the left half of cd, $\angle prq$ decreases, and correspondingly $\alpha_1 + \alpha_2$ increases. Finally, move s to its canonical position.

Case 2: $y_2 \leq 1/2$. (Note that Case 2 covers the entire range of y, both $y \leq 1/2$ and $y \geq 1/2$.) We maximize Λ is two steps. First, we assume that the line ℓ is fixed, and determine optimal positions for r and s. In the second step, we optimize over all positions

of ℓ (subject to the constraint $y_2 \geq 1/2$).
Specifically, we first prove that

$$\alpha_1 + \alpha_2 \leq \frac{\pi}{4} + \arctan\frac{y_1}{1-y_1} \tag{15}$$

$$\alpha_3 + \alpha_4 \leq \frac{\pi}{4} + \arctan\frac{y_2}{1-y_2}. \tag{16}$$

We distinguish two cases depending on whether $\alpha_2 = \angle dpr$ or $\alpha_2 = \angle drp$.

Case 2.1: $\alpha_2 = \angle dpr \leq \pi/4$.] Recall that $y_1 \leq y_2$. In addition, we have $x_2 \leq y_1$ and $1 - y_1 \leq x_1$. We need to show that

$$\arctan\frac{x_2}{y_1} + \arctan\frac{y_1}{x_1} \leq \frac{\pi}{4} + \arctan\frac{y_1}{1-y_1}.$$

Similar to Case 1.1, this inequality is obtained by a continuous movement of r to the right until $|dr| = |dp|$.

Case 2.2: $\boldsymbol{\alpha_2 = \angle drp \leq \pi/4}$. We have $y_1 \leq x_2$ and $x_1 \leq 1 - y_1$. We need to show that

$$\arctan\frac{y_1}{x_2} + \arctan\frac{y_1}{x_1} \leq \frac{\pi}{4} + \arctan\frac{y_1}{1-y_1}.$$

Similar to Case 1.2, this inequality is obtained by a continuous movement of r to the left until $|dr| = |dp|$.
Cases 2.1 and 2.2 together prove inequality (15). The proof of inequality (16) is analogous. To conclude the analysis of case 2 for $y \in [0, 1/2]$, we need to verify that:

$$\left(\frac{\pi}{4} + \arctan\frac{y_1}{1-y_1}\right) + \left(\frac{\pi}{4} + \arctan\frac{y_2}{1-y_2}\right) \leq \frac{\pi}{2} + \arctan\frac{y}{1-y}, \text{ or equivalently,}$$

$$\arctan\frac{y_1}{1-y_1} + \arctan\frac{y_2}{1-y_2} \leq \arctan\frac{y}{1-y}. \tag{17}$$

Applying the tangent function to both sides of the inequality, it remains to show that

$$\frac{\frac{y_1}{1-y_1} + \frac{y_2}{1-y_2}}{1 - \frac{y_1}{1-y_1}\frac{y_2}{1-y_2}} \leq \frac{y}{1-y}, \text{ or equivalently,}$$

$$\frac{y - 2y_1y_2}{1-y} \leq \frac{y}{1-y},$$

which obviously holds; moreover, we have equality in the limit when $y_1 \to 0$ and $y_2 \to y$.
To conclude the analysis of case 2 for $y \in [1/2, 1]$, we need to check that

$$\left(\frac{\pi}{4} + \arctan\frac{y_1}{1-y_1}\right) + \left(\frac{\pi}{4} + \arctan\frac{y_2}{1-y_2}\right) \leq \frac{3\pi}{4} + \arctan\frac{2y-1}{3-2y}, \text{ or equivalently,}$$

$$\arctan\frac{y_1}{1-y_1} + \arctan\frac{y_2}{1-y_2} \leq \frac{\pi}{4} + \arctan\frac{2y-1}{3-2y}. \tag{18}$$

Applying the tangent function to both sides of the inequality, it remains to show that

$$\frac{\frac{y_1}{1-y_1} + \frac{y_2}{1-y_2}}{1 - \frac{y_1}{1-y_1}\frac{y_2}{1-y_2}} \leq \frac{1 + \frac{2y-1}{3-2y}}{1 - \frac{2y-1}{3-2y}}, \text{ or equivalently,}$$

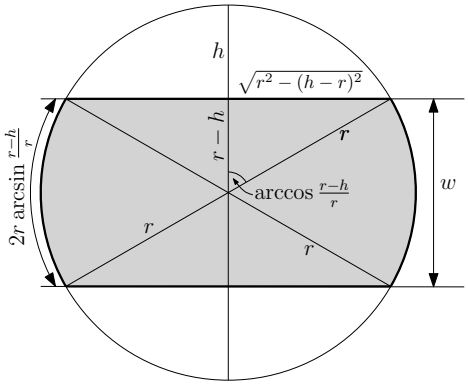


Figure 5 A convex body B of maximum area subject to the constraints that $\mathrm{diam}(B) = 2r$ and the width of B is w.

$$\frac{y - 2y_1y_2}{1-y} \leq \frac{1}{2(1-y)}.$$

The last inequality holds since for $y_1 + y_2 = y$ with the above constraints, the product y_1y_2 is minimized when $y_1 = y - 1/2$ and $y_2 = 1/2$. Moreover, we have equality in the limit when $y_1 \to y - 1/2$ and $y_2 \to 1/2$. This completes the analysis of Case 2, and hence the proof of Lemma 7.

◀

Remark. Note that $f(y)$ is an increasing function of y and $f(1) = 1$; hence $f(y) \leq 1$ for $y \in [0, 1]$, with a strict inequality for $y < 1$. Thus the inequality in Lemma 7 is an improvement of the inequality $\Lambda \leq \pi$ used earlier in (13).

The second key component is establishing an upper bound on the sum of areas $C = \sum_{i=1}^{4} C_i$.

▶ **Lemma 8.** *Define the function*

$$g(y) = 2 - \frac{2y}{\frac{\pi}{2} - \arccos y + y\sqrt{1-y^2}}, \quad y \in [0, 1].$$

Then $2C \leq g(y)\, A$. *This inequality is the best possible for all* $y \in (0, 1]$.

It is known [26, Exercise 6–10] that convex body B of maximum area, subject to the constraints that $\mathrm{diam}(B) = D$ and the width of B is w, is is the intersection of a disk of diameter D and a strip of parallel lines at distance w symmetric about the center of the disk, as shown in Figure 5. Denote by $\Psi(r, h)$ the area of a circular cap of height h in a disk of radius r. An easy calculation [27] yields

$$\Psi(r, h) = r^2 \arccos((r-h)/r) - (r-h)\sqrt{r^2 - (r-h)^2}.$$

Proof of Lemma 8. Recall that P has unit diameter and is enclosed in between two parallel lines at distance y. As noted above, this implies that

$$A \leq \frac{\pi}{4} - 2\Psi\left(\frac{1}{2}, \frac{1-y}{2}\right) = \frac{\pi}{4} - \frac{1}{2}\arccos y + \frac{y}{2}\sqrt{1-y^2}. \tag{19}$$

Consequently, by taking into account that $C = A - y/2$, (19) yields

$$\frac{2C}{A} = \frac{2A-y}{A} = 2 - \frac{y}{A} \leq \frac{2y}{\frac{\pi}{2} - \arccos y + y\sqrt{1-y^2}},$$

as desired. ◀

Remark. Note that $g(y) \leq 1$, for $y \in [0,1]$, with a strict inequality for $y < 1$. Thus the inequality in Lemma 8 is an improvement of the inequality $2C \leq A$ used earlier in (13).

The inequality $\sqrt{\pi A} \leq L/2$, from (10), was used in obtaining the approximating ratio of 1.75. The following lemma refines this inequality in terms of y.

▶ **Lemma 9.** *Define the function*

$$\tau(y) = \frac{1}{4}\sqrt{1 - \frac{\pi^2(1-y)^2}{16(\arcsin y + \sqrt{1-y^2})^2}}, \quad y \in [0,1].$$

Then $\sqrt{\pi A}/(2L) \leq \tau(y)$. *This inequality is the best possible for all* $y \in (0,1]$.

Recall [26, Exercise 6–9] that for every convex body K in the plane, a centrally symmetric convex body K^* is obtained by the symmetrization $K^* = \frac{1}{2}(K - K)$. It is well known that K^* has the same diameter, width and perimeter as K, and that the area of K^* is greater than or equal to that of K; moreover, along every direction, the distance between the two parallel supporting lines of K^* is the same as that for K. This implies that if K has diameter $2r$ and width w, then K^* is contained in a disk or radius r, and in a strip of width w, where the two parallel lines of the strip are equidistant from the center of the disk; see Figure 5.

Proof of Lemma 9. Let $A^* = \text{area}(P^*)$ and $L^* = \text{per}(P^*)$. As noted above, $A \leq A^*$ and $L = L^*$, hence $\sqrt{\pi A}/(2L) \leq \sqrt{\pi A^*}/(2L^*)$. Therefore, it suffices to prove the lemma for P^*.

Let P be a centrally symmetric convex body of diameter 1 and width y. Then the circumradius R of P is $1/2$, and its inradius r is at most $y/2$. Consequently, $R - r \geq (1-y)/2$. Clearly, we have $L \in [2, \pi]$. As noted above, P is contained in the convex body that is the intersection of a disk of diameter 1 and a strip of width y, where the two parallel lines of the strip are equidistant from the disk center (Figure 5). The boundary of this convex body consists of two circular arcs each of length $\arcsin y$, and two line segments each of length $\sqrt{1-y^2}$, and so its perimeter is $2\arcsin y + 2\sqrt{1-y^2}$. Consequently, the perimeter L of P is bounded above as $L \leq 2\arcsin y + 2\sqrt{1-y^2}$.

It is known [22] that for every planar convex body of area A, circumradius R, inradius r, and perimeter L, we have the following sharpening of the first inequality in (10):

$$4\pi A \leq L^2 - \pi^2(R-r)^2.$$

It follows that

$$\frac{\sqrt{\pi A}}{2L} \leq \frac{\sqrt{\frac{L^2}{4} - \frac{\pi^2}{16}(1-y)^2}}{2L} = \frac{1}{4}\sqrt{1 - \frac{\pi^2(1-y)^2}{4L^2}} =: \delta(L,y).$$

Note that for every fixed $y \in [0,1]$, $\delta(L,y)$ is an increasing function in L for $L \in [2,\pi]$. Recall that $L \leq 2\arcsin y + 2\sqrt{1-y^2} \leq \pi$ for $y \in [0,1]$, and by the monotonicity of $\delta(L,y)$ we get

$$\frac{\sqrt{\pi A}}{2L} \leq \frac{1}{4}\sqrt{1 - \frac{\pi^2(1-y)^2}{16(\arcsin y + \sqrt{1-y^2})^2}}$$

for $y \in [0,1]$; consequently, we have $\sqrt{\pi A}/(2L) \leq \tau(y)$, as claimed. ◀

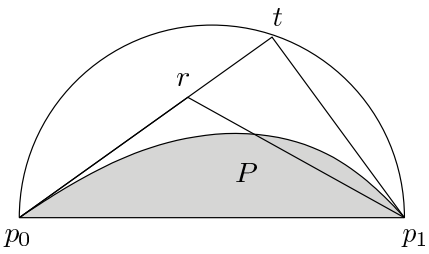


Figure 6 The polygon P lies in a right triangle.

Remark. Note that $\tau(y) \le 1/4$, for $y \in [0,1]$, with a strict inequality for $y < 1$. Thus the inequality in Lemma 9 is an improvement of the inequality $\sqrt{\pi A} \le L/2$ used earlier in (13).

We now finalize the proof of the upper bound in Theorem 1. By using the bounds in Lemmas 7, 8, and 9, we obtain the sharper analogues of (13) and (14):

$$\sqrt{\Lambda \cdot 2C} \le \sqrt{(f(y)\,\pi)(g(y)\,A)} = \sqrt{f(y)g(y)\pi A} \tag{20}$$

$$\rho \le \frac{3L + \sqrt{\Lambda \cdot 2C}}{2L} \le \frac{3L + \sqrt{f(y)g(y)\pi A}}{2L} \le 1.5 + \sqrt{f(y)g(y)}\,\tau(y). \tag{21}$$

A numerical calculation show that $\sqrt{f(y)g(y)}\,\tau(y)$ is maximized to $0.21757\ldots$ at $y = 0.87894\ldots$ and correspondingly $f(y) = 0.92439\ldots$, $g(y) = 0.82244\ldots$, and $\tau(y) = 0.24952\ldots$. Consequently, $\rho \le 1.5 + 0.21757\ldots = 1.71757\ldots < 1.7176$.

The Final Bound on the Approximation Ratio. We refine Lemma 5 using a stronger condition, namely we assume that the polygon $P = (p_0, p_1, \ldots, p_m)$ not only lies in a half-disk of diameter p_0p_1, but also in a right triangle with diameter p_0p_1; refer to Figure 6.

Assume that $P = (p_0, p_1, \ldots, p_m)$ lies in a half-disk of diameter p_0p_1, such that p_0 is at the origin and p_1 is on the positive x-axis. Let $\alpha = \angle p_1p_0p_m < \pi/2$ and $A = \text{area}(P)$. Parametrize the polygonal arc $(p_1, \ldots, p_m)$ in polar coordinate by $(\theta, \lambda(\theta))$ for $\theta \in [0, \alpha]$. Then $\sum_{i=2}^{m} h_i \le \int_0^\alpha \lambda(\theta)\,\mathrm{d}\theta \le \sqrt{2\alpha A}$.

▶ **Lemma 10.** *Assume that $P = (p_0, p_1, \ldots, p_m)$ is contained in a right triangle Δp_0p_1t with $\angle p_0tp_1 = \pi/2$ and $\angle tp_0p_1 = \alpha$. Let $A = \text{area}(P)$. Then $\sum_{i=2}^{m} h_i \le \sqrt{2\mu(\alpha)A} \le \sqrt{2\alpha A}$, where*

$$\mu(\alpha) = \ln^2\left(\frac{1+\sin\alpha}{\cos\alpha}\right) \Big/ \tan\alpha.$$

Proof. Let r be a point on the edge p_0t such that the area of the triangle Δp_0p_1r is A. Observe that to extend the lengths $\lambda(\theta)$ of the triangles of angle $d\theta$ in the fan by the same amount, the triangles corresponding to larger values of β require more area. This implies that, with α, A, and $|p_0p_1|$ fixed, $\int_0^\alpha \lambda(\theta)\,\mathrm{d}\theta$ is maximized when P is exactly the triangle Δp_0p_1r. Moreover, with α and A fixed, and with $|p_0p_1|$ variable, $\int_0^\alpha \lambda(\theta)\,\mathrm{d}\theta$ is maximized when Δp_0p_1r is a right triangle with angle $\angle p_0rp_1 = \pi/2$.

Put $x = |p_0p_1|$ in this extreme case. Then $\lambda(\theta) = x\cos\alpha/\cos\theta$. In the right triangle Δp_0p_1r with $\angle p_0rp_1 = \pi/2$, $2A = |p_0r| \cdot |p_1r| = x\sin\alpha \cdot x\cos\alpha$. Thus $x\cos\alpha = \sqrt{2A/\tan\alpha}$. By integral calculus, we have

$$\int_0^\alpha \frac{\mathrm{d}\theta}{\cos\theta} = \ln\left(\frac{1+\sin\alpha}{\cos\alpha}\right).$$

By Lemma 6 it follows that

$$\sum_{i=2}^{m} h_i \leq \int_0^{\alpha} \lambda(\theta)\,\mathrm{d}\theta = x\cos\alpha \int_0^{\alpha} \frac{\mathrm{d}\theta}{\cos\theta} = \sqrt{2A/\tan\alpha}\,\ln\left(\frac{1+\sin\alpha}{\cos\alpha}\right) = \sqrt{2\mu(\alpha)A}.$$

Moreover, with α and A fixed, (9) holds for $\lambda(\theta) = x\cos\alpha/\cos\theta$ too. Thus $\sqrt{2\mu(\alpha)A} \leq \sqrt{2\alpha A}$. ◀

The function $\mu(\alpha)$ is concave for $0 \leq \alpha < \pi/2$. By Jensen's inequality, the constraint $\sum_{i=1}^{4} \alpha_i \leq f(y)\pi$ implies that $\sum_{i=1}^{4} \mu(\alpha_i) \leq 4\mu(f(y)\pi/4)$. Let $\hat{f}(y) = 4\mu(f(y)\pi/4)/\pi$. Then $\sum_{i=1}^{4} \mu(\alpha_i) \leq \hat{f}(y)\cdot\pi$.

Using $\hat{f}(y)$ instead of $f(y)$ in the final analysis, a numerical calculation show that $\sqrt{f(y)g(y)}\,\tau(y)$ is maximized to $0.21674\ldots$ at $y = 0.87256\ldots$ and correspondingly $\hat{f}(y) = 0.91364\ldots$, $g(y) = 0.82615\ldots$, and $\tau(y) = 0.24947\ldots$. Consequently, $\rho \leq 1.5 + 0.21674\ldots = 1.71674\ldots < 1.7168$, as claimed. ◀

Remark. Note that for a disk Ω of unit radius, every interior barrier must have length at least 2π. Indeed, for every point $p \in \partial\Omega$, blocking p from the line ℓ_p tangent to Ω at p requires that $p \in \Gamma$. It follows that $\partial\Omega \subseteq \Gamma$, which in turn yields $|\Gamma| \geq |\partial\Omega| = 2\pi$, as claimed. On the other hand, a length of 2π clearly suffices. In contrast, using a $(1+\varepsilon)$-interior barrier yields a significant length-reduction, as shown in the following.

▶ **Corollary 11.** *For any $\varepsilon > 0$, the unit disk Ω admits a $(1+\varepsilon)$-interior barrier of length at most $(\pi+2)(1+\varepsilon)$. In particular, the unit disk Ω admits a $(1+\varepsilon)$-interior barrier of length at most 5.1416, provided that $\varepsilon > 0$ is sufficiently small.*

Proof. Assume that Ω is centered at the origin. For a given $\varepsilon > 0$, let n be a sufficiently large even integer such that a regular n-gon P_n inscribed in $(1+\varepsilon)\Omega$ contains Ω, and P_n has a horizontal diameter pq. Consider an interior barrier for P_n that consists of the half of the perimeter of P_n below the x-axis and the heights of a fan triangulation for the remainder of P_n above the x-axis.

We use Lemma 6 for bounding the sum of heights, and for this purpose, we parametrize P_n in polar coordinates with respect to vertex p. Observe that $\lambda(\theta) \leq 2(1+\varepsilon)\cos\theta$, for $\theta \in [0, \frac{\pi}{2}]$. The perimeter of P_n is bounded from above by the corresponding perimeter of $(1+\varepsilon)\Omega$, and the sum of heights is at most $\int_0^{\pi/2} \lambda(\theta)\,\mathrm{d}\theta$ by Lemma 6. Hence the length of this barrier is bounded from above by

$$(1+\varepsilon)\pi + \int_0^{\pi/2} 2(1+\varepsilon)\cos\theta\,\mathrm{d}\theta = (\pi+2)(1+\varepsilon),$$

which is at most 5.1416 when ε is sufficiently small. ◀

4 A Lower Bound on the Approximation Ratio of Shermer's Algorithm

In this section we prove the lower bound given in Theorem 2. Refer to Figure 7.

Let ℓ_S and ℓ_P, respectively, be the lengths of the two barriers illustrated on the left and the right. We have

$$\ell_S = 1 + 2a,$$

and

$$\ell_P(b) = b + 2a(1-b) + 2\sqrt{(a^2-1)b^2 + (2-b)^2}.$$

Figure 7 Two barriers for a rhombus with shorter diagonal of length 2 and with sides of equal length $a \geq 4$. Left: The barrier found by Shermer's procedure. Right: The barrier found by Provan et al. [23]. The height from the top vertex to the dashed line in the right barrier is $b \leq 1$.

The derivative of $\ell_P(b)$ is

$$\ell'_P(b) = 1 - 2a + \frac{(a^2-1)\cdot 2b - 2(2-b)}{\sqrt{(a^2-1)b^2 + (2-b)^2}} = 1 - 2a + \frac{2a^2b - 4}{\sqrt{a^2b^2 - 4b + 4}}.$$

Setting $\ell'_P(b) = 0$ yields

$$(4a^2 - 4a + 1)(a^2b^2 - 4b + 4) = 4a^4b^2 - 16a^2b + 16,$$

which simplifies to

$$a^2b^2 - 4b - 4(4a^2 - 4a - 3)/(4a - 1).$$

For $a \geq 4$, this quadratic equation in b has a unique positive real root

$$b_0 = \frac{4 + \sqrt{16 + 16a^2(4a^2 - 4a - 3)/(4a-1)}}{2a^2} = \frac{2 + 2\sqrt{1 + a^2(4a^2 - 4a - 3)/(4a-1)}}{a^2},$$

and the length $\ell_P(b)$ of the barrier of Provan et al. is minimized when $b = b_0$. In particular, $b_0 = 1$ when $a = 4$, and $b_0 < 1$ when $a > 4$.

Assisted by a computer program, we can verify that for $a \geq 4$, the ratio $\ell_S/\ell_P(b_0)$ is maximized to $1.00769\ldots$ when $a = 16.299\ldots$ ($\sqrt{a^2-1} = 16.268\ldots$), and correspondingly $b_0 = 0.49053\ldots$. ◀

5 Concluding Remarks

Observe that when the input is a regular n-gon P_n, our algorithm returns a barrier whose length is at most $1.5\pi + \int_{\pi/4}^{\pi/2} \cos(\theta)\, \mathrm{d}\theta = 1.5\pi + 2 - \sqrt{2} = 1.6845\ldots\pi$. On the other hand our algorithm attains a ratio at most 1.7176 for *every* convex polygon. This may indicate that its analysis is quite tight.

The length of the barrier constructed in Corollary 11 for the regular n-gon P_n is at most $5.1416 = 1.6367\ldots\pi$. We believe that this barrier is not too far from the optimal one. This may indicate that the algorithm itself is quite good, at least for polygons similar in shape, fat polygons in particular.

All these estimates however are expressed in terms of the same trivial lower bound (1). The reader can notice that for polygons P that are long and skinny, the lower bound $\mathrm{per}(P)/2$ is not too bad. Indeed we indicate below how to modify our algorithm so that it returns an interior barrier whose length is close to $\mathrm{per}(P)/2$. On the other hand, it is worth pointing out that the bottleneck in the analysis of our algorithm is for large values of y, namely $y \approx 0.88$. For such values, we believe that the lower bound $\mathrm{per}(P)/2$ in (1) is quite loose. The conclusion is that further improvement in the approximation ratio of our algorithm for interior barriers relies on an improved lower bound beyond $\mathrm{per}(P)/2$ for fat polygons (i.e., with large y, in our analysis.), and that the case of small y is not too hard to deal with. We conjecture that the approximation ratio of the following algorithm is below 1.1.

Algorithm. In addition to the four candidate interior barriers Γ_i, $i = 1, 2, 3, 4$, defined earlier, we add two new candidates, Γ_+ and Γ_-. The barrier Γ_+ consists of the boundary of P below the diameter pq, the height h_r from r in Δrpq, and the two fans of Shermer heights to the left and to the right of h_r, corresponding to the minimum angles in Δrdp and Δrcq. The barrier Γ_- is defined analogously, by the boundary of P above the diameter pq, etc. The algorithm returns the shortest of these six barriers.

References

1 V. Akman, An algorithm for determining an opaque minimal forest of a convex polygon, *Information Processing Letters* **24** (1987), 193–198.

2 F. Bagemihl, Some opaque subsets of a square, *Michigan Math. J.* **6** (1959), 99–103.

3 H. T. Croft, Curves intersecting certain sets of great-circles on the sphere, *J. London Math. Soc. (2)* **1** (1969), 461–469.

4 H. T. Croft, K. J. Falconer, and R. K. Guy, *Unsolved Problems in Geometry*, Springer, New York, 1991.

5 E. D. Demaine and J. O'Rourke, Open problems from CCCG 2007, in *Proc. 20th Canadian Conference on Computational Geometry* (CCCG 2008), Montréal, Canada, August 2008, pp. 183–190.

6 P. Dublish, An $O(n^3)$ algorithm for finding the minimal opaque forest of a convex polygon, *Information Processing Letters* **29(5)** (1988), 275–276.

7 A. Dumitrescu and M. Jiang, Computational Geometry Column 58, *SIGACT News Bulletin* **44(4)** (2013), 73–78.

8 A. Dumitrescu and M. Jiang, The opaque square, in *Proc. 30th Annual Symposium on Computational Geometry* (SOCG 2014), ACM Press, 2014, pp. 529–538. An earlier version (November 2013) at `arXiv.org/abs/1311.3323v1`.

9 A. Dumitrescu, M. Jiang, and J. Pach, Opaque sets, *Algorithmica* **69** (2014), 315–334.

10 H. G. Eggleston, The maximal in-radius of the convex cover of a plane connected set of given length, *Proc. London Math. Soc. (3)* **45** (1982), 456–478.

11 V. Faber and J. Mycielski, The shortest curve that meets all the lines that meet a convex body, *American Mathematical Monthly* **93** (1986), 796–801.

12 V. Faber, J. Mycielski and P. Pedersen, On the shortest curve which meets all the lines which meet a circle, *Ann. Polon. Math.* **44** (1984), 249–266.

13 S. R. Finch, *Mathematical Constants*, Cambridge University Press, 2003.

14 H. M. S. Gupta and N. C. B. Mazumdar, A note on certain plane sets of points, *Bull. Calcutta Math. Soc.* **47** (1955), 199–201.

15 R. Honsberger, *Mathematical Morsels*, Dolciani Mathematical Expositions, No. 3, The Mathematical Association of America, 1978.

16 R. E. D. Jones, Opaque sets of degree α, *American Mathematical Monthly* **71** (1964), 535–537.

17 A. Kawamura, S. Moriyama, Y. Otachi, and J. Pach, A lower bound on opaque sets, preprint, April 2014; `arXiv:1403.3894v2`.

18 B. Kawohl, Some nonconvex shape optimization problems, in *Optimal Shape Design* (A. Cellina and A. Ornelas, editors), vol. 1740/2000 of Lecture Notes in Mathematics, Springer, 2000.

19 R. Klötzler, Universale Rettungskurven I, *Zeitschrifte für Analysis und ihre Anwendungen* **5** (1986), 27–38.

20 R. Klötzler and S. Pickenhain, Universale Rettungskurven II, *Zeitschrifte für Analysis und ihre Anwendungen* **6** (1987), 363–369.

21 S. Mazurkiewicz, Sur un ensemble fermé, punctiforme, qui rencontre toute droite passant par un certain domaine (Polish, French summary), *Prace Mat.-Fiz.* **27** (1916), 11–16.

22 R. Osserman, The isoperimetric inequality, *Bull. Amer. Math. Soc.* **84** (1978), 1182–1238.

23 J. S. Provan, M. Brazil, D. A. Thomas and J. F. Weng, Minimum opaque covers for polygonal regions, preprint, October 2012, `arXiv:1210.8139v1`.

24 P. R. Scott and P. W. Awyong, Inequalities for convex sets, *Journal of Inequalities in Pure and Applied Mathematics* **1** (2000), article 6.

25 T. Shermer, A counterexample to the algorithms for determining opaque minimal forests, *Information Processing Letters* **40** (1991), 41–42.

26 I. M. Yaglom and V. G. Boltyanskiĭ, *Convex Figures*, Holt, Rinehart and Winston, New York, 1961.

27 D. Zwillinger, *CRC Standard Mathematical Tables and Formulae*, 31st Edition, CRC Press, 2002.

Hardness of Submodular Cost Allocation: Lattice Matching and a Simplex Coloring Conjecture

Alina Ene[*1,2] **and Jan Vondrák**[3]

1 **Center for Computational Intractability, Princeton University**
aene@cs.princeton.edu
2 **Department of Computer Science and DIMAP, University of Warwick**
3 **IBM Almaden Research Center**
jvondrak@us.ibm.com

Abstract

We consider the Minimum Submodular Cost Allocation (MSCA) problem [3]. In this problem, we are given k submodular cost functions $f_1, \ldots, f_k : 2^V \to \mathbb{R}_+$ and the goal is to partition V into k sets $A_1, \ldots, A_k$ so as to minimize the total cost $\sum_{i=1}^{k} f_i(A_i)$. We show that MSCA is inapproximable within any multiplicative factor even in very restricted settings; prior to our work, only Set Cover hardness was known. In light of this negative result, we turn our attention to special cases of the problem. We consider the setting in which each function f_i satisfies $f_i = g_i + h$, where each g_i is monotone submodular and h is (possibly non-monotone) submodular. We give an $O(k \log |V|)$ approximation for this problem. We provide some evidence that a factor of k may be necessary, even in the special case of Hypergraph Labeling [3]. In particular, we formulate a simplex-coloring conjecture that implies a Unique-Games-hardness of $k - 1 - \epsilon$ for k-uniform Hypergraph Labeling and label set $[k]$. We provide a proof of the simplex-coloring conjecture for $k = 3$.[1]

1998 ACM Subject Classification Analysis of Algorithms and Problem Complexity, Optimization

Keywords and phrases Minimum Cost Submodular Allocation, Submodular Optimization, Hypergraph Labeling

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.144

1 Introduction

Labeling problems arise in a number of applications including document classification, image segmentation, facility location, and others. The general problem asks for a labeling of a ground set V by k labels in a way that minimizes a certain notion of "cost"; this cost can penalize "similar elements" being labeled differently, or elements being assigned a label that they "do not prefer". A classical example is the Graph Multiway Cut problem where given a graph $G = (V, E)$ with k terminals $t_1, \ldots, t_k \in V$, we want to partition the vertices into k disjoint sets $S_1, \ldots, S_k$ such that $t_i \in S_i$ and we minimize the number of edges between different parts. Over time, more general versions of this problem have been proposed in order to capture the fact that vertices might have more nuanced (weighted) preferences to be labeled in a certain way [16], relationships more general than pairwise might be present [11],

* Part of this work was done while the author was visiting IBM Almaden. Supported in part by NSF grant CCF-1016684 and a Chirag Foundation graduate fellowship.
[1] Subsequent to this work, a proof of the simplex-coloring conjecture has been found [17].

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 144–159

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

etc. This led to the study of problems such as Metric Labeling [16], 0-Extension [4], and Node-weighted / Hypergraph Multiway Cut [10]. The main object of our study is an abstract version of this problem, the Minimum Submodular Cost Allocation (MSCA) problem, introduced in [3]. In this problem, the cost function associated with each label is a submodular function; $f : 2^V \to \mathbb{R}$ is submodular if $f(A) + f(B) \geq f(A \cap B) + f(A \cup B)$ for every $A, B \subseteq V$.

Minimum Submodular Cost Allocation (MSCA). *Given k submodular functions $f_1, \ldots, f_k : 2^V \to \mathbb{R}_+$, find a partition $(A_1, \ldots, A_k)$ of V that minimizes $\sum_{i=1}^k f_i(A_i)$.*

This captures problems such as Graph Multiway Cut, Hypergraph Multiway Cut, and Uniform Metric Labeling due to the fact that the cut function in graphs and hypergraphs is submodular. However, MSCA is considerably more general than these problems. The special case of the problem in which each of the functions f_i is monotone is equivalent to the Submodular Facility Location problem considered by Svitkina and Tardos [24] who gave an $O(\log |V|)$-approximation and a matching hardness via a reduction from Set Cover. Following previous work, Chekuri and Ene [3, 2] considered several special cases of the MSCA problem in which the functions are non-monotone but they have additional structure. In [3] it was shown that, if each function f_i is the sum of a monotone submodular function g_i and a symmetric submodular function h that is the same for all of the labels, one can achieve an $O(\log |V|)$ approximation. Furthermore, several well-studied partitioning problems such as Graph Multiway Cut and Node-weighted Multiway Cut can be cast as special cases of MSCA in which the functions f_i arise from a single underlying submodular function. Submodular Multiway Partition, a special case of MSCA, was proposed as a unifying umbrella for these problems [25] and shown to admit a $(2 - 2/k)$-approximation which is best possible [2, 7].

Despite this progress, the MSCA problem itself was not very well understood. On the hardness side, prior to our work the best lower bound for the general MSCA problem was the Set Cover hardness shown by Svitkina and Tardos [24] for the special case in which all of the functions are monotone. On the positive side, no approximation guarantees have been known for the MSCA problem with non-monotone functions. This is perhaps surprising, since submodular minimization problems typically admit at least polynomial approximation factors; for example, $O(|V|)$ or $\tilde{O}(\sqrt{|V|})$ approximations are achievable for several minimization problems with submodular costs [14, 12, 23]. One of the main questions left open by previous work was to bridge this wide gap.

Our Results. In this paper, we show that the MSCA problem is in fact inapproximable within *any multiplicative factor* even in very restricted settings. The following theorem formally states our main hardness result.

▶ **Theorem 1.** *It is* **NP***-hard to decide whether the optimal value for a* Minimum Submodular Cost Allocation *problem is zero, even for $k = 3$ and functions of the form $f_i(S) = c_i(S) + \delta_{G_i}(S)$ where $c_i(S) = \sum_{j \in S} c_{ij}$ with 0/1 coefficients c_{ij} and δ_{G_i} is the cut function of a directed graph[2] G_i.*

As an intermediate result, we prove the NP-completeness of two related combinatorial problems that we call Lattice Matching and Partition Matching (see Section 2); we believe this might be of independent interest.

In light of this negative result, we turn our attention to special cases of the problem. In particular, we consider the following problem introduced in [3].

[2] The cut function of $G = (V, A)$ is $\delta_G(S) = |\{(v, w) \in A : v \in S, w \notin S\}|$.

Monotone-restricted MSCA. *For each $i \in [k]$, let $g_i : 2^V \to \mathbb{R}_+$ be a* monotone *submodular function (the "assignment cost). and let $h : 2^V \to \mathbb{R}_+$ be a (possibly non-monotone) submodular function (the "separation cost"). The* Monotone-restricted MSCA *problem is the special case of* MSCA *in which $f_i = g_i + h$ for each $i \in [k]$.*

As mentioned above, [3] gave an $O(\log |V|)$ approximation for a special case where the separation cost function h is *symmetric*. This is best possible even for $h = 0$ which yields the Submodular Facility Location problem [24]. In this paper, we give an $O(k \log |V|)$ approximation for the general Monotone-restricted MSCA problem in which the separation cost function h is not necessarily symmetric.

▶ **Theorem 2.** *There is an $O(k \log |V|)$-approximation for the* Monotone-restricted MSCA *problem.*

Our approach is based on a reduction to the symmetric case via symmetrization of the separation cost h. This result, although quite straightforward, provides us with a very general setting in which the MSCA problem admits a non-trivial approximation. The remaining question is whether the factor of k in Theorem 2 can be eliminated. We provide some evidence that the factor of k may be necessary for the following special case of the problem, introduced in [3].

Hypergraph Labeling. *Given a hypergraph $H = (V, E)$ with edge weights $w(e) \geq 0$ and vertex assignment costs $c(v, i) \geq 0$, find a labeling $\ell : V \to [k]$ so as to minimize $\sum_{v \in V} c(v, \ell(v)) + \sum_{e \in E : |\ell[e]| > 1} w(e)$.*

In other words, we want to minimize the assignment costs of the vertices plus the weight of the edges that are cut (receive multiple labels). This problem is a common generalization of Uniform Metric Labeling [16] and Hypergraph Multiway Cut [18]; i.e., instead of pairwise relationships we consider multi-tuple interactions and we also have modular assignment costs. On the other hand, Hypergraph Labeling can be cast as a special case of Monotone-restricted MSCA (see [3] or Section 4). Building on the work of Kleinberg and Tardos [16] for Uniform Metric Labeling, Chekuri and Ene [3] gave a d-approximation for the Hypergraph Labeling problem, where d is the maximum size of a hyperedge. Our result above gives an $O(k \log |V|)$-approximation for Hypergraph Labeling.

We provide some evidence that a factor of k might be necessary, in particular when $k = d$. We propose a conjecture (somewhat reminiscent of Sperner's Lemma [22]) which implies that a natural LP relaxation of the Hypergraph Labeling problem has integrality gap $k - 1$ for k-uniform hypergraphs and any approximation factor below $k - 1$ would refute the Unique Games Conjecture (using a general result of [7]). We prove the conjecture in the special case of $k = 3$ and thus we obtain an integrality gap and Unique Games hardness of $2 - \epsilon$ for this case; previously, only an integrality gap of $4/3$ was known [16].

Organization. The rest of the paper is organized as follows. In Section 2, we prove the inapproximability of the general MSCA problem, as well as the **NP**-completeness of the related Lattice Matching and Partition Matching problems. In Section 3, we show our approximation result for the Monotone-restricted MSCA problem. In Section 4, we discuss a conjecture that would imply hardness for the Hypergraph Labeling problem. Some details and proofs are deferred to the appendix.

2 Hardness of Minimum Submodular Cost Allocation

If $k = 2$, the MSCA problem can be reduced to submodular function minimization as follows. Let $f : 2^V \to \mathbb{R}_+$ be the function such that $f(S) = f_1(S) + f_2(V \setminus S)$. It is easy to see that f is submodular (Proposition 16). A submodular function can be minimized in polynomial time [5, 19, 21, 13, 15], and therefore MSCA is in **P** when $k = 2$.

The main result of this section is that for $k \geq 3$, the MSCA problem does not admit any finite approximation factor. In particular, we prove that for $k \geq 3$ it is **NP**-hard to decide whether the optimal solution has value zero or nonzero. We use functions in a particular form, using the cut function of a directed graph: $\delta_G(S) = |\{(v, w) \in A(G) : v \in S, w \notin S\}|$.

▶ **Theorem 3.** *It is* **NP***-hard to decide whether the optimal value for an instance of* MSCA *is zero, even for $k = 3$ and functions of the form $f_i(S) = c_i(S) + \delta_{G_i}(S)$ where c_i is a linear function and δ_{G_i} is the cut function of a directed graph G_i.*

We start with the following well-known fact. For any non-negative submodular function, the collection of sets of zero value forms a *Boolean lattice* (see Proposition 14; we recall that a Boolean lattice is a family of sets $\mathcal{L} \subseteq 2^V$ closed under unions and intersections, i.e. $\forall A, B \in \mathcal{L}; A \cap B \in \mathcal{L}, A \cup B \in \mathcal{L}$). This observation suggests that the question of checking for solutions of zero value is related to the following problem that we call Lattice Matching.

Lattice Matching. *Given k Boolean lattices $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_k \subset 2^V$ (by a suitable compact representation), find k disjoint sets $S_1 \in \mathcal{L}_1, \ldots, S_k \in \mathcal{L}_k$ such that $\bigcup_{i=1}^k S_i = V$.*

We show that this problem is **NP**-complete for $k \geq 3$, by a reduction from 1-in-3 SAT [20]. A technical issue is the question of representing a lattice compactly on the input. For this purpose we use a representation of lattices by directed graphs that goes back to Birkhoff [1]. In fact this construction also provides explicit submodular functions whose zeros are exactly the points of the respective lattice, and hence we prove Theorem 3.

A special case of a lattice is the collection of all unions of sets from some partition (collection of disjoint sets) $\mathcal{P}_i$. Thus the Lattice Matching problem has the following natural special case.

Partition Matching. *Given k collections of sets $\mathcal{P}_1, \ldots, \mathcal{P}_k \subset 2^V$, where each $\mathcal{P}_i$ is a partition of a subset of V (the sets in $\mathcal{P}_i$ are disjoint), find disjoint sets $S_1, \ldots, S_\ell \in \bigcup_{i=1}^k \mathcal{P}_i$ such that $\bigcup_{i=1}^\ell S_i = V$.*

Just like the problems above, the Partition Matching problem is in **P** for $k \leq 2$. We prove that this problem is **NP**-complete for $k = 5$. We leave it as an open question whether it is **NP**-complete for $k = 3$ and $k = 4$.

2.1 Representation of Lattices by Directed Graphs

In the following, we describe how to encode a Boolean lattice using a directed graph, and how this connects MSCA and Lattice Matching. We follow the construction of [8].

▶ **Definition 4.** Given a lattice $\mathcal{L} \subseteq 2^V$ such that $\emptyset \in \mathcal{L}$, let $V_\mathcal{L} = \bigcup \{S : S \in \mathcal{L}\}$. For each $v \in V_\mathcal{L}$, let $D(v) = \bigcap \{S : S \in \mathcal{L}, v \in S\}$. We define a directed graph $G_\mathcal{L} = (V_\mathcal{L}, A)$ where $A = \{(v, w) : v \in V_\mathcal{L}, w \in D(v), w \neq v\}$.

The following lemma is implicit in [1, 8]. We include a simple proof for completeness.

▶ **Lemma 5.** *For every lattice $\mathcal{L} \subseteq 2^V$ such that $\emptyset \in \mathcal{L}$, the directed graph $G_\mathcal{L}$ encodes the lattice in the sense that $S \in \mathcal{L}$ if and only if $S \subseteq V_\mathcal{L}$ and $G_\mathcal{L}$ has no arcs from S to $V_\mathcal{L} \setminus S$.*

Proof. If $S \in \mathcal{L}$, then $S \subseteq V_\mathcal{L}$ by the properties of the lattice. Also, for each each $v \in S$, $D(v) = \bigcap\{S' \in \mathcal{L} : v \in S'\} \subseteq S$ and hence all arcs originating at v stay within S.

Conversely, if $S \subseteq V_\mathcal{L}$ and there are no arcs leaving S, we know that for each $v \in S$, $D(v) \subseteq S$. By the properties of lattices, $D(v) = \bigcap\{S' \in \mathcal{L}, v \in S'\} \in \mathcal{L}$. If $S \neq \emptyset$, we get that $S = \bigcup_{v \in S} D(v) \in \mathcal{L}$. If $S = \emptyset$, then $S \in \mathcal{L}$ by assumption. ◀

Thus the directed graph $G_\mathcal{L}$ encodes the lattice $\mathcal{L}$ in a compact way: its description has size $O(n^2)$, where $n = |V|$. Furthermore, we observe that this description provides a submodular function whose zeros are exactly the sets in $\mathcal{L}$.

▶ **Lemma 6.** *For a lattice $\mathcal{L} \subset 2^V$ defined by the directed graph $G_\mathcal{L}$, the following function is submodular and its zeros are exactly the sets in $\mathcal{L}$:*

$$f_\mathcal{L}(S) = |S \setminus V_\mathcal{L}| + \delta_{\mathcal{G}_\mathcal{L}}(S)$$

where $\delta_{\mathcal{G}_\mathcal{L}}(S) = |\{(v, w) \in A(\mathcal{G}_\mathcal{L}) : v \in S, w \notin S\}|$ is the directed cut function of $G_\mathcal{L}$.

Proof. By Lemma 5, $S \in \mathcal{L}$ if and only if $S \subseteq V_\mathcal{L}$ and there are no arcs from S to outside of S in $G_\mathcal{L}$, which occurs if and only if $f_\mathcal{L}(S) = 0$. ◀

It follows that the Lattice Matching problem – where the lattices are given by its associated directed graph – is equivalent to checking whether the MSCA instance in which the functions are $\{f_{\mathcal{L}_i} \mid i \in [k]\}$ has zero cost. To prove Theorem 3, it remains to prove the **NP**-completeness of Lattice Matching under this encoding.

2.2 NP-completeness of Partition Matching and Lattice Matching

In this section, we prove that the Lattice Matching problem is **NP**-complete for $k = 3$. First, as a warm-up, let us prove that its special case, the Partition Matching problem, is **NP**-complete for $k = 7$. We reduce from the following **NP**-complete problem [9].

3-bounded 3-set Packing. *Given a system of triples $\mathcal{T} \subseteq 2^V$ such that each element of V is contained in at most 3 triples, it is **NP**-complete to decide whether there exists a collection of disjoint triples covering V.*

▶ **Theorem 7.** Partition Matching *is **NP**-complete for $k = 7$.*

Proof. Given an instance of 3-bounded 3-set Packing, we observe that each triple intersects at most 6 other triples (2 for each of its elements). Thus we can inductively color the triples with 7 colors in such a way that intersecting triples get different colors. We define $\mathcal{P}_i$ to be the collection of all triples of color i. We obtain an instance of Partition Matching with $k = 7$, for which it is **NP**-complete to decide whether there exists a collection of disjoint triples covering V. ◀

For lower values of k, we use more careful reductions from the Monotone 1-in-3 SAT problem [20].

Monotone 1-in-3 SAT. *Given a formula $\bigwedge_{i=1}^{m}(x_{i_1} \vee x_{i_2} \vee x_{i_3})$ (without negations), it is* **NP***-complete to find a Boolean assignment such that in each clause $(x_{i_1} \vee x_{i_2} \vee x_{i_3})$, exactly one variable is True and two variables are False.*

▶ **Theorem 8.** Partition Matching *is* **NP***-complete for $k = 5$.*

Proof. Given an instance of Monotone 1-in-3 SAT, we produce an instance of Partition Matching as follows. We define a ground set V consisting of

- An element v_j for each variable x_j.
- Two elements $x_j^i, \neg x_j^i$ for each occurrence of a variable x_j in clause i.

On this ground set, we define the following 5 collections of sets:

- $\mathcal{P}_1$ contains for each variable x_j a set $\{v_j\} \cup \{x_j^i : \forall$clause i containing variable $x_j\}$.
- $\mathcal{P}_2$ contains for each variable x_j a set $\{v_j\} \cup \{\neg x_j^i : \forall$clause i containing variable $x_j\}$.
- $\mathcal{P}_3$ contains for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ a set $\{x_{i_1}^i, \neg x_{i_2}^i, \neg x_{i_3}^i\}$.
- $\mathcal{P}_4$ contains for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ a set $\{\neg x_{i_1}^i, x_{i_2}^i, \neg x_{i_3}^i\}$.
- $\mathcal{P}_5$ contains for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ a set $\{\neg x_{i_1}^i, \neg x_{i_2}^i, x_{i_3}^i\}$.

We call the sets in $\mathcal{P}_1 \cup \mathcal{P}_2$ *variable-assignment* sets, and the sets in $\mathcal{P}_3 \cup \mathcal{P}_4 \cup \mathcal{P}_5$ *clause-assignment* sets. Observe that in each collection $\mathcal{P}_i$, the sets are pairwise disjoint. I.e., we have an instance of Partition Matching with $k = 5$.

If there is a Boolean assignment such that exactly one variable is each clause is satisfied, we produce a solution of Partition Matching as follows. For each variable $x_j =$ True, we choose the variable-assignment set containing v_j that is in $\mathcal{P}_2$ (i.e. containing all the elements $\neg x_j^i$). For each variable $x_j =$ False, we choose the variable-assignment set containing v_j that is in $\mathcal{P}_1$ (i.e. containing all the elements x_j^i). Finally, for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$, we choose the set corresponding to its assignment, from either $\mathcal{P}_3$, $\mathcal{P}_4$ or $\mathcal{P}_5$. It is easy to verify that these sets are disjoint and cover the entire ground set V.

Conversely, let us assume that there is a collection of disjoint sets $\mathcal{F} \subset \mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3 \cup \mathcal{P}_4 \cup \mathcal{P}_5$ that covers the ground set V. Since $\mathcal{F}$ must cover the element v_j for each variable, it must contain a variable-assignment set in either $\mathcal{P}_1$ or $\mathcal{P}_2$ (no other sets contain v_j). This choice determines a Boolean assignment: we set $x_j =$ True if v_j is covered by a set from $\mathcal{P}_2$, and $x_j =$ False if v_j is covered by a set from $\mathcal{P}_1$. Now, consider the 6 elements $x_{i_1}^i, \neg x_{i_1}^i, x_{i_2}^i, \neg x_{i_2}^i, x_{i_3}^i, \neg x_{i_3}^i$ for clause i. Exactly 3 of these elements are covered by sets from $\mathcal{P}_1$ and $\mathcal{P}_2$, hence the remaining 3 elements must form a clause-assignment set in $\mathcal{P}_3 \cup \mathcal{P}_4 \cup \mathcal{P}_5$. These clause-assignment sets correspond to satisfying assignments and hence the formula is satisfied by the Boolean assignment that we defined. ◀

Finally, we prove that Lattice Matching is NP-complete for $k = 3$. Note that here we use the full flexibility of the Lattice Matching problem; we do not know whether the Partition Matching problem is NP-complete for $k = 3$.

▶ **Theorem 9.** Lattice Matching *is* **NP***-complete for $k = 3$.*

Proof. We use the same reduction as in the proof of Theorem 8, but we combine the 5 partitions into 3 lattices. Specifically, using the notation from the proof above, we define

- $\mathcal{L}_1 = \mathrm{cl}(\mathcal{P}_1 \cup \mathcal{P}_3)$
- $\mathcal{L}_2 = \mathrm{cl}(\mathcal{P}_2)$
- $\mathcal{L}_3 = \mathrm{cl}(\mathcal{P}_4 \cup \mathcal{P}_5)$

where $\mathrm{cl}(\mathcal{P})$ means all the sets that can be generated from $\mathcal{P}$ by taking unions and intersections. By construction, $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ are lattices that contain all the sets that were contained in $\mathcal{P}_1, \ldots, \mathcal{P}_5$ (as well as some additional sets). In the case of a satisfiable formula, we can still

choose disjoint sets covering V as above. Let S_i be the union of those of these sets that are contained in $\mathcal{L}_i$ (i.e., S_i is also in $\mathcal{L}_i$), and $S_1 \cup S_2 \cup S_3 = V$ is a feasible solution of the Lattice Matching problem. The potential issue with this construction is that we might have created a feasible solution of the Lattice Matching problem in case the formula is not satisfiable. Let us argue that this is not the case.

For each variable x_j, the element v_j is contained only in the variable-assignment sets arising from $\mathcal{P}_1$ and $\mathcal{P}_2$, and sets formed by unions of these variable-assignment sets with other sets in $\mathcal{L}_1 = \mathrm{cl}(\mathcal{P}_1 \cup \mathcal{P}_3), \mathcal{L}_2 = \mathrm{cl}(\mathcal{P}_2)$. Recall that these two variable assignment sets appear in $\mathcal{P}_1, \mathcal{P}_2$ respectively, and so their union/intersection is not part of the lattices that we generate. Therefore, v_j must be covered by a set that was generated from one of the two variable-assignment sets for x_j; depending on which one is used, we set x_j to True (if the set $\{v_j, \neg x_j^i, \neg x_j^{i'}, \ldots\}$ is used) or False (if the set $\{v_j, x_j^i, x_j^{i'}, \ldots\}$ is used).

By our construction of the three lattices, for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$, some of the respective elements $x_{i_1}^i, x_{i_2}^i, x_{i_3}^i, \neg x_{i_1}^i, \neg x_{i_2}^i, \neg x_{i_3}^i$ are going to appear as singletons in a lattice. In $\mathcal{L}_1$, we get new sets obtained by intersecting sets in $\mathcal{P}_1$ and $\mathcal{P}_3$: Specifically, this is the singleton $\{x_{i_1}^i\}$ for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ (and sets obtained by taking unions of these singletons with other sets in $\mathcal{P}_1 \cup \mathcal{P}_3$). In $\mathcal{L}_2$, we obtain only the unions of sets in $\mathcal{P}_2$ (which are disjoint). In $\mathcal{L}_3$, we obtain by intersection the singleton $\{\neg x_{i_1}^i\}$ for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$, and again sets obtained by unions with other sets in $\mathcal{P}_4 \cup \mathcal{P}_5$ (recall the definitions of $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \mathcal{P}_4$ and $\mathcal{P}_5$).

Observe that the construction is not symmetric with respect to the three elements of a clause such as $x_{i_1}^i \vee x_{i_2}^i \vee x_{i_3}^i$. While $\{x_{i_1}^i\}$ and $\{\neg x_{i_1}^i\}$ appear as sets in $\mathcal{L}_1, \mathcal{L}_3$ respectively, $\{x_{i_2}^i\}$ and $\{\neg x_{i_2}^i\}$ do not appear as singletons in any lattice. This is because $x_{i_2}^i$ appears in exactly one set in $\mathcal{P}_4$, and $\neg x_{i_2}^i$ appears in exactly one set in $\mathcal{P}_3$ and one set in $\mathcal{P}_5$. Thus we do not form any intersections that can produce $\{x_{i_2}^i\}$ or $\{\neg x_{i_2}^i\}$. By the same argument, we do not form any intersections that can produce a pair containing $x_{i_2}^i$ or $\neg x_{i_2}^i$. Every set in $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ that contains $x_{i_2}^i$ or $\neg x_{i_2}^i$ contains either the variable-assignment set $\{v_j, \ldots, \}$, or a triple corresponding to a satisfying assignment of the i-th clause, e.g. $\{x_{i_1}^i, \neg x_{i_2}^i, \neg x_{i_3}^i\}$. If x_{i_2} was set to True, then $\neg x_{i_2}^i$ is covered by a variable-assignment set but $x_{i_2}^i$ is not because that would cause v_j to be covered twice. Therefore, the only way that $x_{i_2}^i$ can be covered is by a triple corresponding to a satisfying assignment of the i-th clause. By the same argument, this assignment is consistent with our assignment of variables. We can repeat this argument for each clause; it proves that if there is a feasible solution of the Lattice Matching problem, then our assignment satisfies every clause of the formula. ◀

3 Monotone-restricted MSCA Algorithm

In the previous section, we showed that the MSCA problem does not admit any multiplicative approximation whatsoever. This can be viewed as evidence that MSCA is "not the right generalization" of problems like Multiway Cut, Uniform Metric Labeling, etc. In this section we restrict MSCA to some extent, so that we still obtain a fairly general partitioning problem but one that allows some non-trivial approximation. We consider the Monotone-restricted MSCA problem in which each assignment cost function g_i is an arbitrary monotone submodular function and the separation cost function h is an arbitrary submodular function (but the same one for all label values). We seek a partitioning (or labeling) $(S_1, S_2, \ldots, S_k)$ minimizing $\sum_{i=1}^{k}(g_i(S_i) + h(S_i))$.

We observe that, for any submodular function h, $h'(S) = h(S) + h(V \setminus S)$ is a symmetric submodular function (see Proposition 17 in the appendix). Since h' is symmetric, we can use

the algorithm of [3] to construct a labeling for the instance of Monotone-restricted MSCA in which the assignment costs are given by g_i and the separation cost is given by h'. For any labeling, we can relate its h' cost to its h cost as follows.

▶ **Proposition 10.** *Let $(A_1, \ldots, A_k)$ be a labeling. We have*

$$\sum_{i=1}^{k} h(A_i) \leq \sum_{i=1}^{k} h'(A_i) \leq k \sum_{i=1}^{k} h(A_i).$$

Proof. The first inequality follows from the fact that h is non-negative. Therefore it suffices to show the second inequality. A non-negative submodular function is sub-additive and thus we have

$$h(V \setminus A_i) = h(\cup_{j \neq i} A_j) \leq \sum_{j \neq i} h(A_j).$$

Therefore $\sum_{i=1}^{k} h(V \setminus A_i) \leq (k-1) \sum_{i=1}^{k} h(A_i)$ and $\sum_{i=1}^{k} h'(A_i) \leq k \sum_{i=1}^{k} h(A_i)$. ◀

Let OPT and OPT$'$ be the costs of the optimal solution for the original instance in which the separation cost function is h and the modified instance in which the separation cost function is h', respectively. By the above, OPT$' \leq k \cdot$ OPT. Let $(A_1, \ldots, A_k)$ be the solution constructed by the algorithm of [3] for the modified instance. The result of [3] is that there is an $O(\log |V|)$-approximation for Monotone-restricted MSCA whenever the functions g_i are monotone submodular and h is symmetric submodular. It follows that

$$\sum_{i=1}^{k} g_i(A_i) + \sum_{i=1}^{k} h(A_i) \leq \sum_{i=1}^{k} g_i(A_i) + \sum_{i=1}^{k} h'(A_i) \leq O(\log |V|)\text{OPT}' \leq O(k \log |V|)\text{OPT}.$$

Therefore we have the following theorem.

▶ **Theorem 11.** *There is an $O(k \log |V|)$-approximation for the* Monotone-restricted MSCA *problem.*

We remark that the factor of $\log |V|$ is necessary due to the hardness of the Submodular Facility Location problem, which is the case of $h = 0$. The same hardness can be obtained when h is a simple symmetric submodular function and the g_i's are modular functions – see Appendix B.

4 Hypergraph Labeling and Sperner's Colorings

In this section, we consider the Hypergraph Labeling problem, which is a special case of Monotone-restricted MSCA (see Section 1 for a definition). As we have shown in the previous section, Monotone-restricted MSCA (and thus Hypergraph Labeling) admits an $O(k \log |V|)$ approximation, where k is the number of labels. Also, the Hypergraph Labeling problem was shown to admit a d-approximation when the size of each hyperedge is at most d [3]. We provide some evidence that a factor of k might be necessary for this problem, in particular when $d = k$.

4.1 LP Relaxations for Hypergraph Labeling

Chekuri and Ene [3] gave a convex-programming relaxation (LE-Rel) for MSCA that is based on the Lovász extension of a submodular function. Let us review the convex relaxation LE-Rel in the special case of the Hypergraph Labeling problem. In LE-Rel, we have variables $x_{v,i}$ for

$v \in V, i \in [k]$. Recall that the objective function in Hypergraph Labeling can be modeled as $\sum_{i=1}^{k} f_i(S_i)$ where $f_i(S) = g_i(S) + h(S)$, $g_i(S) = \sum_{v \in S} c(v,i)$ and

$$h(S) = \sum_{e \in E : r(e) \in S, e \not\subseteq S} w(e)$$

is the rooted version of the hypergraph cut function, for some choice of a root $r(e) \in e$ for every $e \in E$.

The LE-Rel relaxation is based on the Lovász extension of the objective functions $f_i(S)$. By linearity, the Lovász extension $\hat{f}_i(\mathbf{x})$ can be written as $\hat{f}_i(\mathbf{x}) = \hat{g}_i(\mathbf{x}) + \hat{h}(\mathbf{x}) = \sum_{v \in V} c(v,i) x_{v,i} + \hat{h}(\mathbf{x})$ since the function g_i is linear. The Lovász extension of the rooted hypergraph cut function h can be written as follows: for a uniformly random threshold $\lambda \in [0,1]$,

$$\hat{h}(\mathbf{x}) = \sum_{e \in E} w(e) \Pr[x_{r(e)} > \lambda \ \& \ \exists v \in e; x_v \leq \lambda] = \sum_{e \in E} w(e)(x_{r(e)} - \min_{v \in e} x_v)$$

(see [6] for details). The objective function of LE-Rel is $\sum_{i=1}^{k} \hat{f}_i(\mathbf{x}_i) = \sum_{i=1}^{k} (\hat{g}_i(\mathbf{x}_i) + \hat{h}(\mathbf{x}_i))$, where $\mathbf{x}_i \in \mathbb{R}^V$ is the assignment vector for label i. We note that $\sum_{i=1}^{k} x_{r(e),i} = 1$ by the assignment constraint in LE-Rel, and hence we have

$$\sum_{i=1}^{k} \hat{h}(\mathbf{x}_i) = \sum_{e \in E} w(e) \left(1 - \sum_{i=1}^{k} \min_{v \in e} x_{v,i}\right).$$

Hence we can write the full LE-Rel relaxation for Hypergraph Labeling as follows.

LE-Rel for Hypergraph Labeling

$$\min \quad \sum_{v \in V} \sum_{i=1}^{k} c(v,i) x_{v,i} + \sum_{e \in E} w(e) \left(1 - \sum_{i=1}^{k} \min_{v \in e} x_{v,i}\right) :$$

$$\sum_{i=1}^{k} x_{v,i} = 1 \qquad \forall v \in V$$

$$x_{v,i} \geq 0 \qquad \forall v \in V, i \in [k]$$

Formally, this is not in the form of a linear program but it is easy to see that the expression $\min_{v \in e} x_{v,i}$ can be replaced by a new variable $z_{e,i}$ with constraints $z_{e,i} \leq x_{v,i} \forall v \in e$. We prefer to keep the form above for compactness.

Next, we observe that this LP is equivalent to the "Local Distribution LP" considered in [7]. In the Local Distribution LP, we have $x_{v,i}$ variables as above, and also $y_{e,\alpha}$ variables for each hyperedge $e \in E$ and each possible assignment $\alpha \in [k]^e$. The hyperedge variables $y_{e,\alpha}$ can be interpreted as a distribution over labelings of the respective hyperedge e. The hyperedge variables must be consistent with the vertex variables in the sense that all assignments such that $\alpha_v = i$ should add up to $\sum_{\alpha \in [k]^e : \alpha_v = i} y_{e,\alpha} = x_{v,i}$. The Local Distribution LP reads as follows.

Local Distribution LP for Hypergraph Labeling

$$\min \quad \sum_{v\in V}\sum_{i=1}^{k} c(v,i)x_{v,i} + \sum_{e\in E,\alpha\neq(\ell,\ell,\ldots,\ell)} w(e)y_{e,\alpha} :$$

$$\sum_{\alpha\in[k]^e,\alpha_v=i} y_{e,\alpha} = x_{v,i} \qquad \forall v\in e\in E, i\in[k]$$

$$\sum_{i=1}^{k} x_{v,i} = 1 \qquad \forall v\in V$$

$$x_{v,i}, y_{e,\alpha} \geq 0 \qquad \forall v,i,e,\alpha$$

Consider a feasible assignment to the variables $x_{v,i}$. Given this assignment, the Local Distribution LP aims to minimize the cut cost $\sum_{\alpha\neq(\ell,\ell,\ldots,\ell)} y_{e,\alpha}$ for each hyperedge e, subject to the condition $\sum_{\alpha\in[k]^e;\alpha_v=i} y_{e,\alpha} = x_{v,i}$. We claim that the optimal way to do this is to set $y_{e,(i,i,\ldots,i)} = \min_{v\in e} x_{v,i}$ for each $i\in[k]$, and then distribute the remaining mass $1-\sum_{i=1}^{k}\min_{v\in e} x_{v,i}$ among variables $y_{e,\alpha}$ where α contains more than 1 label, so as to satisfy the consistency constraints $\sum_{\alpha\in[k]^e,\alpha_v=i} y_{e,\alpha} = x_{v,i}$. This is possible to do greedily, since as long as we have $\sum_{\alpha\in[k]^e} y_{e,\alpha} < 1$, there is some label for each vertex such that $\sum_{\alpha\in[k]^e,\alpha_v=i} y_{e,\alpha} < x_{v,i}$, and so we can increase the variable $y_{e,\alpha}$ for the corresponding assignment. This achieves the objective value of $\sum_{v\in V}\sum_{i=1}^{k} c(v,i)x_{v,i} + \sum_{e\in E} w(e)(1-\sum_{i=1}^{k}\min_{v\in e} x_{v,i})$, identical to that of LE-Rel. On the other hand, $\min_{v\in e} x_{v,i}$ is the maximum value that we can assign to $y_{e,(i,i,\ldots,i)}$ without violating the consistency constraints, so the contribution of hyperedge e cannot be lower than $1-\sum_{i=1}^{k}\min_{v\in e} x_{v,i}$, just like in LE-Rel.

The work of [7] implies that for any variant of Min CSP including the Not-Equal predicate (which is the case here), it is Unique-Games-hard to achieve any approximation better than the integrality gap of the Local Distribution LP. Therefore, the LP presented here (in either equivalent form) is in some sense the optimal tool to consider when developing approximation algorithms for the Hypergraph Labeling problem.

4.2 A Simplex Coloring Conjecture

In this section, we describe a conjecture that would imply an integrality gap close to $k-1$ for the k-uniform Hypergraph Labeling problem with label set $[k]$.

Sperner's Simplex example. Let $q\geq 1$ be an integer and consider the $(k-1)$-dimensional simplex defined by

$$\Delta = \left\{\mathbf{x}=(x_1,x_2,\ldots,x_k)\in\mathbb{R}^k : \mathbf{x}\geq 0, \sum_{i=1}^{k} x_i = q\right\}.$$

We consider a vertex set of all the points in Δ with integer coordinates:

$$V = \left\{\mathbf{a}=(a_1,a_2,\ldots,a_k)\in\mathbb{Z}^k : \mathbf{a}\geq 0, \sum_{i=1}^{k} a_i = q\right\}.$$

We define an (unweighted) k-uniform hypergraph $H=(V,E)$ on this vertex set whose hyperedges are indexed by $\mathbf{b}\in\mathbb{Z}_+^k$ such that $\sum_{i=1}^{k} b_i = q-1$: we have

$$E = \left\{e(\mathbf{b}) : \mathbf{b}=(b_1,b_2,\ldots,b_k)\in\mathbb{Z}^k, \mathbf{b}\geq 0, \sum_{i=1}^{k} b_i = q-1\right\}$$

where

$$e(\mathbf{b}) = \{(b_1+1, b_2, \ldots, b_k), (b_1, b_2+1, \ldots, b_k), \ldots, (b_1, b_2, \ldots, b_k+1)\}.$$

For each vertex $\mathbf{a} \in V$, we have a list of admissible labels $L(\mathbf{a})$, which is

$$L(\mathbf{a}) = \{i \in [k] : a_i > 0\}.$$

Formally, in the setting of the Hypergraph Labeling problem, we define the assignment cost to be $c(\mathbf{a}, i) = 0$ whenever $i \in L(\mathbf{a})$ and $c(\mathbf{a}, i) = \infty$ otherwise. We also define the edge weights to be $w(e) = 1$ for all $e \in E$.

We call a labeling $\ell : V \to [k]$ Sperner-admissible if $\ell(\mathbf{a}) \in L(\mathbf{a})$ for each $\mathbf{a} \in V$. The reader may notice that this is a restriction identical to the framework of Sperner's Lemma [22], where the points on each lower-dimensional face can be labeled only with colors corresponding to vertices of that face. The conclusion of Sperner's Lemma is that there must exist a cell (a scaled copy of the simplex Δ) whose vertices have all k colors. We remark that this cell might not be a member of E since E consists only of scaled copies of Δ *without* rotation. Nevertheless, we need a different statement here.

▶ **Conjecture 12.** *For any Sperner-admissible labeling $\ell : V \to [k]$, there are at least $\binom{q+k-3}{k-2}$ hyperedges $e \in E$ that are not monochromatic under ℓ.*

Let us comment on where the expression $\binom{q+k-3}{k-2}$ comes from. The total number of hyperedges in E is the number of partitions of $q-1$ into a sum of k nonnegative integers. By a well-known combinatorial argument, this is equal to the number of choices of $k-1$ barriers from among $(q-1)+(k-1)$ points and barriers, which is $|E| = \binom{q+k-2}{k-1}$. Similarly, the number of hyperedges that are adjacent to a given facet of the simplex (e.g. those satisfying $b_k = 0$) is equal to the number of hyperedges in a similarly defined $(k-2)$-dimensional simplex, which is $\binom{q+k-3}{k-2}$. We conjecture that the labeling minimizing the number of non-monochromatic hyperedges is one that labels all vertices $\mathbf{a}$ with $a_1 > 0$ by label 1, and then it labels all vertices with $a_1 = 0$ arbitrarily (subject to the restrictions given above). Under this labeling, all the hyperedges $e(\mathbf{b})$ such that $b_1 > 0$ are labeled monochromatically by 1. The only hyperedges that receive more than 1 label are those where $b_1 = 0$, and the number of such hyperedges is $\binom{q+k-3}{k-2}$ as we argued above.

Implications for the Integrality Gap. Let us see what this conjecture would imply for the Hypergraph Labeling problem. We can view the geometric description above naturally as an LP solution for the Hypergraph Labeling problem where vertex $\mathbf{a}$ is mapped to $\mathbf{x_a} = \frac{1}{q}\mathbf{a}$. By construction, for each point $\mathbf{x_a}$ the nonzero coordinates are exactly those in the admissible list $L(\mathbf{a})$, so the assignment cost of this LP solution is $\sum_{v \in V} c(v,i)x_{v,i} = 0$. To compute the cut cost, consider a single hyperedge $e(\mathbf{b})$. The contribution of this hyperedge is

$$1 - \sum_{i=1}^{k} \min_{v \in e(\mathbf{b})} x_{v,i} = 1 - \sum_{i=1}^{k} \frac{1}{q} b_i = \frac{1}{q}$$

since we have $\sum_{i=1}^{k} b_i = q-1$ for every hyperedge $e(\mathbf{b})$. Thus, each hyperedge contributes $\frac{1}{q}$ to the LP cost and in total we have

$$LP = \frac{1}{q}|E| = \frac{1}{q}\binom{q+k-2}{k-1}.$$

In contrast, Conjecture 12 implies that for any feasible solution, at least $\binom{q+k-3}{k-2}$ hyperedges must be non-monochromatic and hence $OPT \geq \binom{q+k-3}{k-2}$. For $q \to \infty$, we obtain

$$\frac{OPT}{LP} = \frac{\binom{q+k-3}{k-2}}{\frac{1}{q}\binom{q+k-2}{k-1}} = \frac{q}{q+k-2}(k-1) \to k-1.$$

So Conjecture 12 implies that the integrality gap can be arbitrarily close to $k-1$ and consequently any approximation algorithm with a factor below $k-1$ would refute the Unique Games Conjecture [7].

4.3 Proof of Conjecture 12 for $k = 3$

▶ **Proposition 13.** *In the example above for $k = 3$ and $q \geq 1$, for any Sperner-admissible labeling $\ell : V \to [3]$ there are at least q triangles in E that are not monochromatic.*

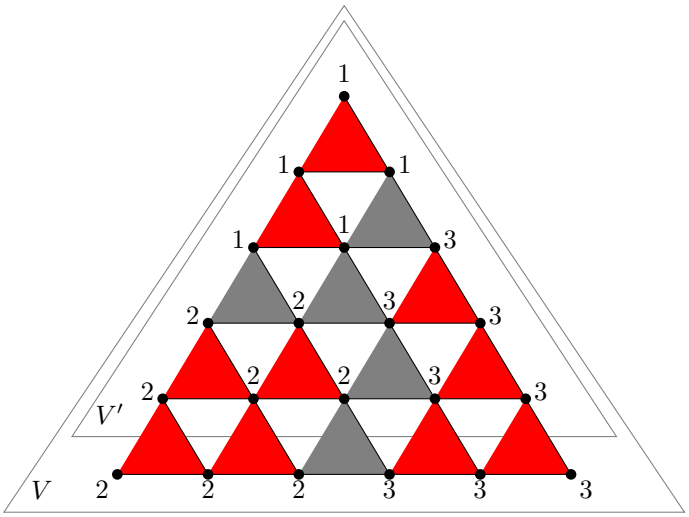


Figure 1 A Sperner-admissible coloring for $k = 3$ and $q = 5$. The set E of hyperedges consists of the shaded triangles. At least $q = 5$ triangles must be non-monochromatic (in gray). The sets V and V' are as in the proof.

Proof. We prove this statement by induction on q. If $q = 1$, then we have only one hyperedge in E which is required to be labeled $(1, 2, 3)$, so the statement holds.

If $q > 1$, let $V' = V \setminus \{\mathbf{a} \in V : a_1 = 0\}$ and $E' = E \cap \binom{V'}{3}$. Observe that $H' = (V', E')$ is the same hypergraph that our construction gives for $q - 1$, but the labeling restrictions are somewhat different. In particular, in V the bottom row (vertices with $a_1 = 0$) is allowed to be labeled only by colors $\{2, 3\}$, while in V' the bottom row (vertices with $a_1 = 1$) can be labeled by all 3 colors.

Consider any labeling $\ell : V \to [3]$. We distinguish two cases. If there is no vertex $\mathbf{a} \in V'$ such that $a_1 = 1$ and $\ell(\mathbf{a}) = 1$, then the labeling on V' is Sperner-admissible and we can apply induction to $H' = (V', E')$. The inductive hypothesis says that there are at least $q - 1$ non-monochromatic triangles in E'. Moreover, the bottom row of V is colored $\{2, 3\}$ and its endpoints have different colors. Therefore, there is an edge in the bottom row which is labeled $\{2, 3\}$. This gives 1 additional non-monochromatic triangle in $E \setminus E'$, for a total of q non-monochromatic triangles in E.

The remaining case is that there are some vertices in the bottom row of V' ($a_1 = 1$), labeled $\ell(\mathbf{a}) = 1$. Let the number of such vertices be s. Let us define a modified coloring $\ell' : V' \to \{2, 3\}$ where

- $\ell'(\mathbf{a}) = \ell(\mathbf{a})$ if $a_1 > 1$ or $\ell(\mathbf{a}) \neq 1$,
- $\ell'(\mathbf{a}) = 2$ if $\ell(\mathbf{a}) = 1$, $a_1 = 1$ and $a_2 > 0$,
- $\ell'(\mathbf{a}) = 3$ if $\ell(\mathbf{a}) = 1$, $a_1 = 1$ and $a_2 = 0$ (which implies $a_3 = q - a_1 - a_2 > 0$).

To summarize, starting from ℓ, we changed s labels in the bottom row of V' from 1 to 2 or 3. How many non-monochromatic triangles could we have added this way? The only new non-monochromatic triangles under ℓ' are those that were labeled $(1, 1, 1)$ under ℓ and they were adjacent to the bottom row. There could have been at most $s - 1$ such triangles, because each of them contains two vertices of the bottom row and the leftmost and rightmost vertex labeled 1 can appear only once in such a triangle. Therefore, we added at most $s - 1$ non-monochromatic triangles under ℓ' compared to ℓ.

Now, ℓ' is a Sperner-admissible labeling of $H' = (V', E')$. By the inductive hypothesis, there are at least $q - 1$ non-monochromatic triangles in E' under ℓ'. Consequently, there are at least $(q - 1) - (s - 1) = q - s$ non-monochromatic triangles in E' under the labeling ℓ. In addition, there are s non-monochromatic triangles in $E \setminus E'$, since every vertex labeled 1 in the bottom row of V' contributes one such triangle (its neighbors in the bottom row of V cannot be labeled 1). Therefore, there are at least q non-monochromatic triangles in E under ℓ. ◀

References

1 G. Birkhoff. Rings of sets. *Duke Mathematical Journal*, 3:443–454, 1937.

2 C. Chekuri and A. Ene. Approximation algorithms for submodular multiway partition. In *Proc. of IEEE FOCS*, 2011.

3 C. Chekuri and A. Ene. Submodular cost allocation problem and applications. In *Proc. of ICALP*, pages 354–366, 2011.

4 G. Călinescu, H. J. Karloff, and Y. Rabani. Approximation algorithms for the 0-extension problem. In *Proc. of ACM-SIAM SODA*, 2001.

5 W. H. Cunningham. On submodular function minimization. *Combinatorica*, 5(3):185–192, 1985.

6 A. Ene. Approximation algorithms for submodular optimization and graph problems. Ph.D. thesis, University of Illinois, Urbana-Champaign, 2013.

7 A. Ene, J. Vondrák, and Y. Wu. Local distribution and the symmetry gap: Approximability of multiway partitioning problems. In *Proc. of ACM-SIAM SODA*, pages 306–325, 2013.

8 S. Fujishige. *Submodular functions and optimization.* Elsevier Science, 2005.

9 M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, 1979.

10 N. Garg, V. V. Vazirani, and M. Yannakakis. Multiway cuts in node weighted graphs. *Journal of Algorithms*, 50(1):49–61, 2004.

11 D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamical systems. *VLDB Journal*, 8(3-4):222–236, 2000.

12 G. Goel, C. Karande, P. Tripathi, and L. Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. In *Proc. of IEEE FOCS*, pages 755–764, 2009.

13 S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions. In *Proc. of ACM STOC*, pages 97–106, 2000.

14 S. Iwata and K. Nagano. Submodular function minimization under covering constraints. In *Proc. of IEEE FOCS*, pages 671–680, 2009.

15 S. Iwata and J. B. Orlin. A simple combinatorial algorithm for submodular function minimization. In *Proc. of ACM-SIAM SODA*, pages 1230–1237, 2009.

16 J. M. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. *Journal of the ACM*, 49(5):616–639, 2002.

17 M. Mirzakhani, 2014. personal communication.

18 K. Okumoto, T. Fukunaga, and H. Nagamochi. Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. *Algorithmica*, pages 1–20, 2010.

19 M. Queyranne. A combinatorial algorithm for minimizing symmetric submodular functions. In *Proc. of ACM-SIAM SODA*, pages 98–101, 1995.

20 T. J. Schaefer. The complexity of satisfiability problems. In *Proc. of ACM STOC*, pages 216–226, 1978.

21 A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.

22 E. Sperner. Neuer Beweis für die Invarianz der Dimensionszahl und des Gebietes. *Math. Sem. Univ. Hamburg*, 6:265–272, 1928.

23 Z. Svitkina and L. Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. In *Proc. of IEEE FOCS*, pages 697–706, 2008.

24 Z. Svitkina and E. Tardos. Facility location with hierarchical facility costs. *ACM Transactions on Algorithms*, 6(2):1–22, 2010.

25 L. Zhao, H. Nagamochi, and T. Ibaraki. Greedy splitting algorithms for approximating multiway partition problems. *Mathematical Programming*, 102(1):167–183, 2005.

A Basic Lemmas

The following lemmas are folklore. We include the proofs for completeness.

▶ **Proposition 14.** *For every non-negative submodular function $f : 2^V \to \mathbb{R}_+$, the set $\mathcal{L} = \{S \subseteq V : f(S) = 0\}$ forms a lattice, i.e., it is closed under unions and intersections.*

Proof. Let $f(A) = f(B) = 0$. Then $f(A \cup B) + f(A \cap B) \leq f(A) + f(B) = 0$, and f is nonnegative, so we must have $f(A \cup B) = f(A \cap B) = 0$ as well. ◀

▶ **Proposition 15.** *Let $f : 2^V \to \mathbb{R}$ be a submodular function. Let $g : 2^V \to \mathbb{R}$ be the function such that $g(S) = f(V \setminus S)$ for all $S \subseteq V$. Then g is also submodular.*

Proof. Consider any two sets A and B. Using the submodularity of f, we have

$$\begin{aligned} g(A) + g(B) &= f(V \setminus A) + f(V \setminus B) \\ &\geq f((V \setminus A) \cap (V \setminus B)) + f((V \setminus A) \cup (V \setminus B)) \\ &= f(V \setminus (A \cup B)) + f(V \setminus (A \cap B)) \\ &= g(A \cup B) + g(A \cap B). \end{aligned}$$

◀

▶ **Proposition 16.** *Let $f_1, f_2 : 2^V \to \mathbb{R}$ be submodular functions. Then $f(S) = f_1(S) + f_2(V \setminus S)$ is also submodular.*

Proof. By Proposition 15, $g_2(S) = f_2(V \setminus S)$ is a submodular function. Hence, $f(S) = f_1(S) + g_2(S)$ is also submodular. ◀

▶ **Proposition 17.** *Let $f : 2^V \to \mathbb{R}$ be a submodular function. Let f' be the following function: $f'(S) = f(S) + f(V \setminus S)$ for each set $S \subseteq V$. Then f' is submodular and symmetric.*

Proof. We have $f'(V \setminus S) = f(V \setminus S) + f(S) = f'(S)$ so f' is symmetric. By Proposition 15, f' is also submodular. ◀

B Hardness of Monotone-restricted MSCA

In this section, we show that Monotone-restricted MSCA is Set Cover hard even if the assignment cost functions g_i are modular and the separation cost function h is symmetric. This shows that the factor of $\log n$ in Theorem 11 is necessary. In fact, for the special case of Monotone-restricted MSCA where the separation cost function h is symmetric submodular, it was already known that an $O(\log n)$-approximation can be achieved [3].

▶ **Theorem 18.** *There is an approximation preserving reduction from the* Set Cover *problem to the special case of* Monotone-restricted MSCA *in which each assignment cost function g_i is modular and the separation cost function h is symmetric. Moreover, each function g_i satisfies $g_i(S) = \sum_{v \in S} c(v, i)$, where $c(v, i)$ is either zero or infinity. The function h satisfies $h(A) = 0$ if $A \in \{\emptyset, V\}$ and $h(A) = 1$ otherwise.*

▶ Remark. The function $h : 2^V \to \mathbb{R}$ that satisfies $h(A) = 0$ if $A \in \{\emptyset, V\}$ and $h(A) = 1$ otherwise, is the cut function of a hypergraph on the vertex set V that has a single hyperedge containing all the vertices. This function is known to be symmetric submodular, which is easy to verify directly as well.

Our reduction is based on the reduction of Svitkina and Tardos [24] for Monotone MSCA. Consider an instance of Set Cover consisting of a set $V = \{v_1, \ldots, v_n\}$ of n elements and a collection $\mathcal{S} = \{S_1, \ldots, S_k\}$ of k sets. We construct an instance of Monotone-restricted MSCA as follows. The ground set is the set V of elements in the Set Cover instance. We have a label i for each set S_i in $\mathcal{S}$. For each element v and each label i, we have an assignment cost $c(v, i)$ that is equal to zero if $v \in S_i$ and ∞ otherwise. The assignment cost function g_i for the i-th label is defined as follows: $g_i(A) = \sum_{v \in A} c(v, i)$ for each set $A \subseteq V$. The separation function h is defined as above: $h(A) = 0$ if $A \in \{\emptyset, V\}$ and $h(A) = 1$ otherwise.

Note that we may assume that there does not exist a set in $\mathcal{S}$ that covers all the elements, since otherwise the solution consisting of such a set is an optimal solution (and this does not happen in hard instances of Set Cover).

▶ **Lemma 19.** *Suppose that there does not exist a set in $\mathcal{S}$ that covers all the elements. Then the* Set Cover *instance has a solution consisting of t sets if and only if the* Monotone-restricted MSCA *instance has a solution of cost t.*

Proof. Consider a solution $\mathcal{S}' \subseteq \mathcal{S}$ for the Set Cover instance. We construct a labeling $A_1, \ldots, A_k$ inductively as follows. We let $A_1 = S_1$ if $S_1 \in \mathcal{S}'$ and $A_1 = \emptyset$ otherwise. Consider an index $i \geq 2$. We let $A_i = S_i \setminus (A_1 \cup \cdots \cup A_{i-1})$ if $S_i \in \mathcal{S}'$ and $A_i = \emptyset$ otherwise.

Note that the resulting sets $A_1, \ldots, A_k$ are disjoint and they cover all the elements. Since $A_i \subseteq S_i$, we have $c(v, i) = 0$ for each $v \in A_i$ and thus $g_i(A_i) = 0$. Additionally, $h(A_i) = 1$ only if $S_i \in \mathcal{S}'$. Therefore the total separation cost of the labeling is at most $|\mathcal{S}'|$.

Conversely, consider a solution $A_1, \ldots, A_k$ for the Monotone-restricted MSCA instance. Note that we may assume that the solution has finite cost and thus $g_i(A_i) = 0$ for all labels i. It follows that $A_i \subseteq S_i$ for each i. We construct a set cover $\mathcal{S}'$ as follows. For each i such that A_i is non-empty, we add S_i to $\mathcal{S}'$. Since the sets A_i cover all the elements and $A_i \subseteq S_i$

for each i, the sets of $\mathcal{S}'$ cover all the elements as well. Since $V \notin \mathcal{S}$, $A_i \neq V$ for all i. Thus the cost of the labeling is equal to the number of non-empty sets in the labeling, which in turn it is equal to $|\mathcal{S}'|$. ◀

Constrained Monotone Function Maximization and the Supermodular Degree*

Moran Feldman[1] and Rani Izsak[2]

1 **School of Computer and Communications, EPFL**
Route Cantonale, 1015 Lausanne, Switzerland
moran.feldman@epfl.ch

2 **Faculty of Mathematics and Computer Science, Weizmann Institute of Science**
234 Herzl Street, Rehovot 7610001, Israel
ran.izsak@weizmann.ac.il

Abstract

The problem of maximizing a constrained monotone set function has many practical applications and generalizes many combinatorial problems such as k-Coverage, Max-SAT, Set Packing, Maximum Independent Set and Welfare Maximization. Unfortunately, it is generally not possible to maximize a monotone set function up to an acceptable approximation ratio, even subject to simple constraints. One highly studied approach to cope with this hardness is to restrict the set function, for example, by requiring it to be submodular. An outstanding disadvantage of imposing such a restriction on the set function is that no result is implied for set functions deviating from the restriction, even slightly. A more flexible approach, studied by Feige and Izsak [ITCS 2013], is to design an approximation algorithm whose approximation ratio depends on the complexity of the instance, as measured by some complexity measure. Specifically, they introduced a complexity measure called supermodular degree, measuring deviation from submodularity, and designed an algorithm for the welfare maximization problem with an approximation ratio that depends on this measure.

In this work, we give the first (to the best of our knowledge) algorithm for maximizing an *arbitrary* monotone set function, subject to a k-extendible system. This class of constraints captures, for example, the intersection of k-matroids (note that a single matroid constraint is sufficient to capture the welfare maximization problem). Our approximation ratio deteriorates gracefully with the complexity of the set function and k. Our work can be seen as generalizing both the classic result of Fisher, Nemhauser and Wolsey [Mathematical Programming Study 1978], for maximizing a submodular set function subject to a k-extendible system, and the result of Feige and Izsak for the welfare maximization problem. Moreover, when our algorithm is applied to each one of these simpler cases, it obtains the same approximation ratio as of the respective original work. That is, the generalization does not incur any penalty. Finally, we also consider the less general problem of maximizing a monotone set function subject to a uniform matroid constraint, and give a somewhat better approximation ratio for it.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases supermodular degree, set function, submodular, matroid, extendible system

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.160

* Work of Moran Feldman is supported in part by ERC Starting Grant 335288-OptApprox.
Work of Rani Izsak is supported in part by the Israel Science Foundation (grant No. 621/12) and by the I-CORE Program of the Planning and Budgeting Committee and the Israel Science Foundation (grant No. 4/11).

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 160–175

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

1 Introduction

A set function f is a function assigning a non-negative real value to every subset of a given ground set $\mathcal{N}$. A set function is (non-decreasing) monotone if $f(A) \leq f(B)$ whenever $A \subseteq B \subseteq \mathcal{N}$. Monotone set functions are often used to represent utility/cost functions in economics and algorithmic game theory. From a theoretical perspective, many combinatorial problems such as k-Coverage, Max-SAT, Set Packing and Maximum Independent Set can be represented as constrained maximization of monotone set functions.

Unfortunately, it is generally not possible to maximize a general monotone set function up to an acceptable approximation ratio, even subject to simple constraints. For example, consider the case of a partition matroid constraint, where the ground set is partitioned into subsets of size m, and we are allowed to pick only a single element from each subset. This problem generalizes the well-known welfare maximization problem[1], and thus, cannot be generally approximated by a factor of $O(\log m/m)$, in time polynomial in n and m (see Blumrosen and Nisan [2]).[2]

One highly studied approach to cope with this hardness is to restrict the set function. A common restriction is submodularity. A set function is submodular if the marginal contribution of an element to a set can only decrease as the set increases. More formally, for every two sets $A \subseteq B \subseteq \mathcal{N}$ and element $u \in \mathcal{N} \setminus B$, $f(B \cup \{u\}) - f(B) \leq f(A \cup \{u\}) - f(A)$. Submodular functions are motivated by many real world applications since they represent the principle of economy of scale, and are also induced by many natural combinatorial structures (*e. g.*, the cut function of a graph is submodular). Fortunately, it has been shown that submodular functions can be maximized, up to a constant approximation ratio, subject to various constraints. For example, maximizing a monotone submodular function subject to the partition matroid constraint, considered above, has a $(1 - 1/e)$-approximation algorithm (see Calinescu, Chekuri, Pal and Vondrák [4]).

An outstanding disadvantage of imposing a restriction on the set function, such as submodularity, is that no result is implied for functions deviating from the restriction, even slightly. A more flexible approach, studied by [12], is to define a complexity measure for set functions, and then design an approximation algorithm whose guarantee depends on this measure. More specifically, [12] introduced a complexity measure called supermodular degree. A submodular set function has a supermodular degree of 0. The supermodular degree becomes larger as the function deviates from submodularity. Feige and Izsak [12] designed a $(1/(d+2))$-approximation algorithm for the welfare maximization problem, where d is the maximum supermodular degree of the bidders' utility functions.

In a classic work, Fisher, Nemhauser and Wolsey [16] introduced a $(1/(k+1))$-approximation algorithm for maximizing a submodular set function subject to a k-extendible system (in fact, they proved this approximation ratio even for a more general class of constraints called k-systems). In this work, we leverage their work, together with the supermodular degree, and give the first (to the best of our knowledge) algorithm for maximizing an *arbitrary* monotone set function subject to a k-extendible system. Note that k-extendible system generalizes, for example, the intersection of k-matroids (see Section 2 for definitions), and thus, also the welfare maximization problem, which can be captured by a single matroid

[1] The welfare maximization problem consists of a set $\mathcal{B}$ of m bidders and a set $\mathcal{N}$ of n items. Each bidder $b \in \mathcal{B}$ has a monotone utility function $u_b : 2^{\mathcal{N}} \to \mathbb{R}^+$. The objective is to assign a disjoint set $\mathcal{N}_b \subseteq \mathcal{N}$ of items to each bidder in a way maximizing $\sum_{b \in \mathcal{N}} u_b(\mathcal{N}_b)$ (*i. e.*, the "social welfare").

[2] This result applies to value oracles, which we use throughout this work.

constraint. As in the works of Fisher, Nemhauser and Wolsey [16] and [12], our algorithm is greedy. Like in [12], the approximation ratio of our algorithm deteriorates gracefully with the complexity of the set function. Interestingly, when our algorithm is applied to the simpler cases studied by [16] and [12], its approximation ratio is exactly the same as that proved by the respective work. That is, we have no penalty for generality, either for handling an arbitrary set function (as opposed to only submodular) or for handling an arbitrary k-extendible system (as opposed to only welfare maximization). We also show an hardness result, depending on k and the supermodular degree of the instance, suggesting the approximation ratio of our algorithm is almost the best possible. Finally, we consider the less general problem of maximizing a monotone set function subject to a uniform matroid constraint (see Section 2), and give a somewhat better approximation ratio for it.

1.1 Related Work

Extensive work has been conducted in recent years in the area of maximizing monotone submodular set functions subject to various constraints. We mention here the most relevant results. Historically, one of the very first problems examined was maximizing a monotone submodular set function subject to a matroid constraint. Several special cases of matroids and submodular functions were studied in [6, 20, 21, 25, 26], using the greedy approach. Recently, the general problem, with an arbitrary matroid and an arbitrary submodular set function, was given a tight approximation of $(1 - 1/e)$ by Calinescu et al. [4]. A matching lower bound is due to [29, 30].

The problem of maximizing a monotone submodular set function over the intersection of k matroids was considered by Fisher et al. [16], who gave a greedy algorithm with an approximation ratio of $1/(k + 1)$, and stated that their proof extends to the more general class of k-systems using the outline of Jenkyns [25] (the extended proof is explicitly given by Calinescu et al. [4]). For k-intersection systems and k-exchange systems, this result was improved by Lee et al. [27] and Feldman et al. [15], respectively, to $1/(k + \varepsilon)$, for every constant $\varepsilon > 0$. The improvement is based on a local search approach that exploits exchange properties of the underlying combinatorial structure. Ward [34] further improved the approximation ratio for k-exchange systems to $2/(k + 3 + \varepsilon)$ using a non-oblivious local search. However, for maximizing a monotone submodular set function over k-extendible independence systems (and the more general class of k-systems), the current best known approximation is still $1/(k + 1)$ [16].

Other related lines of work deal with maximization of non-monotone submodular set functions (constrained or unconstrained) (see [3, 14, 33] for a few examples) and minimization of submodular set functions [17, 18, 23, 24].

The welfare maximization problem (or combinatorial auction) is unique in the sense that it was studied in the context of many classes of utility (set) functions, including classes generalizing submodular set functions such as sub-additive [10] and fractionally sub-additive valuations [9]. For many of these classes a constant approximation algorithm is known [1, 8, 10, 13, 19] assuming access to a demand oracle, which given a vector of prices returns a set of elements maximizing the welfare of a player given these prices. However, when only a value oracle is available to the algorithm (*i. e.*, the only access the algorithm has to the utility functions is by evaluating them on a chosen set) one cannot get a better than a polynomial approximation ratio, even for fractionally sub-additive valuations [9]. We are not aware of any other maximization subject to a constraint problem that was studied with respect to a non-submodular objective before our work.

2 Preliminaries

In this work, we consider set functions $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ that are (non-decreasing) monotone (*i. e.*, $A \subseteq B \subseteq \mathcal{N}$ implies $f(A) \leq f(B)$) and non-negative. We denote the cardinality of $\mathcal{N}$ by n. For readability, given a set $S \subseteq \mathcal{N}$ and an element $u \in \mathcal{N}$ we use $S + u$ to denote $S \cup \{u\}$ and $S - u$ to denote $S \setminus \{u\}$.

2.1 Independence Systems

Given a ground set $\mathcal{N}$, a pair $(\mathcal{N}, \mathcal{I})$ is called an independence system if $\mathcal{I} \subseteq 2^{\mathcal{N}}$ is hereditary (that is, for every set $S \in \mathcal{I}$, every set $S' \subseteq S$ is also in $\mathcal{I}$). Independence systems are further divided into a few known classes. The probably most highly researched class of independence systems is the class of matroids.

▶ **Definition 1** (Matroid)**.** An independence system is a matroid if for every two sets $S, T \in \mathcal{I}$ such that $|S| > |T|$, there exists an element $u \in S \setminus T$, such that $T + u \in \mathcal{I}$. This property is called the *augmentation property* of matroids.

Two important types of matroids are uniform and partition matroids. In a uniform matroid a subset is independent if and only if its size is at most k, for some fixed k. In a partition matroid, the ground set $\mathcal{N}$ is partitioned into multiple subsets $\mathcal{N}_1, \mathcal{N}_2, \ldots, \mathcal{N}_k$, and an independent set is allowed to contain at most a single element from each subset $\mathcal{N}_i$.

Some classes of independence systems are parametrized by a value $k \in \mathbb{N}$ ($k \geq 1$). The following is a simple example of such a class.

▶ **Definition 2** (k-intersection)**.** An independence system $(\mathcal{N}, \mathcal{I})$ is a ***k*-intersection** if there exist k matroids $(\mathcal{N}, \mathcal{I}_1) \ldots (\mathcal{N}, \mathcal{I}_k)$ such that a set $S \subseteq \mathcal{N}$ is in $\mathcal{I}$ if and only if $S \in \bigcap_{i=1}^{k} \mathcal{I}_i$.

The problem of k-dimensional matching can be represented as maximizing a linear function over a k-intersection independence system. In this problem, one looks for a maximum weight matching in a k-sided hypergraph, *i. e.*, an hypergraph where the nodes can be partitioned into k "sides" and each edge contains exactly one node of each side. The representation of this problem as the intersection of k partition matroids consists of one matroid per "side" of the hypergraph. The ground set of such a matroid is the set of edges, and a subset of edges is independent if and only if no two edges in it share a common vertex of the side in question.

The following definition, introduced by Mestre [28], describes a more general class of independence systems which is central to our work.

▶ **Definition 3** (k-extendible)**.** An independence system $(\mathcal{N}, \mathcal{I})$ is a ***k*-extendible** system if for every two subsets $T \subseteq S \in \mathcal{I}$ and element $u \notin T$ for which $T \cup \{u\} \in \mathcal{I}$, there exists a subset $Y \subseteq S \setminus T$ of cardinality at most k for which $S \setminus Y + u \in \mathcal{I}$.

The problem of maximizing a linear function over a k-extendible system captures the problem of k-set packing.[3] In this problem, one is given a weighted collection of subsets of $\mathcal{N}$, each of cardinality at most k, and seeks a maximum weight sub-collection of pairwise disjoint sets. The corresponding k-extendible system is as follows. The ground set contains the sets as elements. The independent subsets are all subsets of pairwise disjoint sets. Let us explain why this is a k-extendible system. Adding a set S of size k to an independent set I, while respecting disjointness, requires that every elements of S is not contained in any other

[3] k-set packing is, in fact, already captured by a smaller class called k-exchange, defined by [15].

set of I. On the other hand, since I is independent, each element is contained in at most one set of I. Therefore, in order to add S, while preserving disjointness, we need to remove up to k sets from I, as required by Definition 3.

The most general class of independence systems considered is given by Definition 5. The following definition is used to define it.

▶ **Definition 4** (Base)**.** Given an independence system $(\mathcal{N}, \mathcal{I})$ and a set $S \subseteq \mathcal{N}$, we say that a set $B \subseteq S$ is a base of S if $B \in \mathcal{I}$ but $B + u \notin \mathcal{I}$ for every element $u \in S \setminus B$. Furthermore, if $S = \mathcal{N}$, then we say that B is a base of the set system itself, or simply, a base.

▶ **Definition 5** (k-system)**.** An independence system $(\mathcal{N}, \mathcal{I})$ is a ***k*-system** if for every set $S \subseteq \mathcal{N}$, the ratio between the sizes of the smallest and largest bases of S is at most k.

An example of a natural problem which can be represented by a k-system, but not by a k-extendible system is given by [4]. The following (strict) inclusions can be shown to hold [4]:

$$\text{matroids} \subset k\text{-intersection} \subset k\text{-extendible systems} \subset k\text{-systems} \ .$$

2.2 Degrees of Dependency

We use the following standard definition.

▶ **Definition 6** (Marginal set function)**.** Let $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ be a set function and let $u \in \mathcal{N}$. The marginal set function of f with respect to u, denoted by $f(u \mid \cdot)$ is defined as $f(u \mid S) \stackrel{\text{def}}{=} f(S+u) - f(u)$. When the underlying set function f is clear from the context, we sometimes call $f(u \mid S)$ the marginal contribution of u to the set S. For subsets $S, T \subseteq \mathcal{N}$, we also use the notation $f(T \mid S) \stackrel{\text{def}}{=} f(S \cup T) - f(S)$.

We recall the definitions of the complexity measures used in this work (defined by [12]).

▶ **Definition 7** (Dependency degree)**.** The dependency degree of an element $u \in \mathcal{N}$ by f is defined as the cardinality of the set $\mathcal{D}_f(u) = \{v \in \mathcal{N} \mid \exists_{S \subseteq \mathcal{N}} f(u \mid S + v) \neq f(u \mid S)\}$, containing all elements whose existence in a set might affect the marginal contribution of u. $\mathcal{D}_f(u)$ is called the dependency set of u by f. The dependency degree of a function f, denoted by $\mathcal{D}_f$, is simply the maximum dependency degree of any element $u \in \mathcal{N}$. Formally, $\mathcal{D}_f = \max_{u \in \mathcal{N}} |\mathcal{D}_f(u)|$. When the underlying set function is clear from the context, we sometimes omit it from the notations.

Note that $0 \leq \mathcal{D}_f \leq n - 1$ for any set function f. $\mathcal{D}_f = 0$ when f is *linear*, and becomes larger as f deviates from linearity.

▶ **Definition 8** (Supermodular (dependency) degree)**.** The supermodular degree of an element $u \in \mathcal{N}$ by f is defined as the cardinality of the set $\mathcal{D}_f^+(u) = \{v \in \mathcal{N} \mid \exists_{S \subseteq \mathcal{N}} f(u \mid S+v) > f(u \mid S)\}$, containing all elements whose existence in a set might *increase* the marginal contribution of u. $\mathcal{D}_f^+(u)$ is called the supermodular dependency set of u by f. The supermodular degree of a function f, denoted by $\mathcal{D}_f^+$, is simply the maximum supermodular degree of any element $u \in \mathcal{N}$. Formally, $\mathcal{D}_f^+ = \max_{u \in \mathcal{N}} |\mathcal{D}_f^+(u)|$. Again, when the underlying set function is clear from the context, we sometimes omit it from the notations.

Note that $0 \leq \mathcal{D}_f^+ \leq \mathcal{D}_f \leq n - 1$ for any set function f. $\mathcal{D}_f^+ = 0$ when f is *submodular*, and becomes larger as f deviates from submodularity.

2.3 Representing the Input

Generally speaking, a set function might assign 2^n different values for the subsets of a ground set of size n. Thus, one cannot assume that any set function has a succinct (*i. e.*, polynomial in n) representation. Therefore, it is a common practice to assume access to a set function via oracles. That is, an algorithm handling a set function often gets an access to an oracle that answers queries about the function, instead of getting an explicit representation of the function. Arguably, the most basic type of an oracle is the value oracle, which given any subset of the ground set, returns the value assigned to it by the set function. Formally:

▶ **Definition 9** (Value oracle)**.** Value oracle of a set function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ is the following:
Input: A subset $S \subseteq \mathcal{N}$.
Output: $f(S)$.

Similarly, since in a given independence system the number of independence subsets might be, generally, exponential in the size of the ground set, it is common to use the following type of oracle.

▶ **Definition 10** (Independence oracle)**.** Independence oracle of an independence system $(\mathcal{N}, \mathcal{I})$ is the following:
Input: A subset $S \subseteq \mathcal{N}$.
Output: A Boolean value indicating whether $S \in \mathcal{I}$.

Our algorithms use the above standard oracles. Additionally, in order to manipulate a function with respect to the dependency/supermodular degree, we need a way to know what are the (supermodular) dependencies of a given element in the ground set. Oracles doing so were introduced by [12], and were used in their algorithms for the welfare maximization problem. Formally:

▶ **Definition 11** (Dependency and Supermodular oracles)**.** Dependency oracle (Supermodular oracle) of a set function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ is the following:
Input: An element $u \in \mathcal{N}$.
Output: The set $\mathcal{D}(u)$ ($\mathcal{D}^+(u)$) of the (supermodular) dependencies of u with respect to f.

2.4 Our Results

Our main result is an algorithm for maximizing any monotone set function subject to a k-extendible system, with an approximation ratio that degrades gracefully as the supermodular degree increases. Note that our algorithm achieves the best known approximation ratios also for the more specific problems of welfare maximization [12] and maximizing a monotone submodular function subject to a k-extendible system [16].

▶ **Theorem 12.** *There exists a* $(1/(k(\mathcal{D}_f^+ + 1) + 1))$*-approximation algorithm of* $\texttt{Poly}(|\mathcal{N}|, 2^{\mathcal{D}_f^+})$ *time complexity for the problem of maximizing a non-negative monotone set function* f *subject to a* k*-extendible system.*

Note that an exponential dependence in $\mathcal{D}_f^+$ is unavoidable, since, otherwise, we would get a polynomial time $(n-1)$-approximation algorithm for maximizing any set function subject to a k-extendible system.[4]

[4] To see that this cannot be done, consider the problem of maximizing the following family of set functions subject to a uniform $k = n/2$ matroid constraint. Each function in the family has a value of 1 for sets

We show a similar result also for the dependency degree, providing a better approximation ratio when $\mathcal{D}_f = \mathcal{D}_f^+$.

▶ **Theorem 13.** *There exists a* $(1/(k(\mathcal{D}_f+1)))$*-approximation algorithm of* $\texttt{Poly}(|\mathcal{N}|, 2^{\mathcal{D}_f})$ *time complexity for the problem of maximizing a non-negative monotone set function* f *subject to a* k*-extendible system.*

On the other hand, we give tight examples for both algorithms guaranteed by Theorems 12 and 13, and show the following hardness result via a reduction from k-dimensional matching.

▶ **Theorem 14.** *No polynomial time algorithm for maximizing a non-negative monotone set function* f *subject to a* k*-intersection independence system has an approximation ratio within* $O\left(\frac{\log k + \log \mathcal{D}_f}{k\mathcal{D}_f}\right)$*, unless* $\mathcal{P} = \mathcal{NP}$*. This is true even if* k *and* $\mathcal{D}_f$ *are considered constants.*

Note that since $\mathcal{D}_f^+ \leq \mathcal{D}_f$ for any set function f, the hardness claimed in Theorem 14 holds also in terms of $\mathcal{D}_f^+$.

Finally, we also consider the special case of a uniform matroid constraint, *i. e.*, where one is allowed to pick an arbitrary subset of $\mathcal{N}$ of size at most k. For this simpler constraint we present an algorithm whose approximation ratio has a somewhat better dependence on $\mathcal{D}_f^+$.[5]

▶ **Theorem 15.** *There exists a* $(1 - e^{-1/(\mathcal{D}_f^+ + 1)})$*-approximation algorithm of* $\texttt{Poly}(|\mathcal{N}|, 2^{\mathcal{D}_f^+})$ *time complexity for the problem of maximizing a non-negative monotone set function* f *subject to a uniform matroid constraint.*

▶ **Theorem 16.** *No polynomial time algorithm for maximizing a non-negative monotone set function* f *subject to a uniform matroid constraint has a constant approximation ratio, unless SSE (Small-Set Expansion Hypothesis)*[6] *is false.*

3 k-Extendible System

In this section we prove Theorems 12 and 14. The proof of Theorem 13 uses similar ideas and is omitted, due to space constraints.

3.1 Algorithm for k-extendible System (Proof of Theorem 12)

We consider in this section Algorithm 1, and prove it fulfills the guarantees of Theorem 12.

First, let us give some intuition. Let APX be an approximate solution and let OPT be an arbitrary optimal solution. Originally, before the algorithm adds any elements to APX, it still can be that it chooses to add all elements of OPT (together) to APX, and get an optimal solution. At each iteration, when adding elements to APX, this possibility might get ruined for some elements of OPT. If we want to keep the invariant that all elements of

strictly larger than k and for a single set A of size k. For all other sets the function assigns the value of 0 (observe that $\mathcal{D}^+(u) = \mathcal{N} - u$ for every element $u \in \mathcal{N}$, hence, the supermodular oracle is useless in this example). Given a random member of the above family, a deterministic algorithm using a polynomial number of oracle queries can determine A only with an exponentially diminishing probability, and thus, will also output a set of value 1 with such an exponentially diminishing probability. Using Yao's principle, this implies an hardness also for randomized algorithms.

[5] The guarantee of Theorem 15 is indeed an improvement over the guarantee of Theorem 13 for 1-extendible system, because for every $x \geq 1$, $1 - e^{-1/x} \geq 1 - (1 - x^{-1} + x^{-2}/2) = x^{-1}(1 - x^{-1}/2) \geq x^{-1}/(1 + x^{-1}) = 1/(x+1)$.

[6] See [31, 32] and Section 4 for definitions and more information about SSE.

Algorithm 1: Extendible System Greedy$(f, \mathcal{I})$

Initialize: $S_0 \leftarrow \varnothing$, $i \leftarrow 0$.
while S_i *is not a base* **do**
  $i \leftarrow i + 1$.
  Let $u_i \in \mathcal{N} \setminus S_{i-1}$ and $D^+_{best}(u_i) \subseteq \mathcal{D}^+(u_i)$ be a pair of an element and a set maximizing $f(D^+_{best}(u_i) + u_i \mid S_{i-1})$ among the pairs obeying $S_{i-1} \cup D^+_{best}(u_i) + u_i \in \mathcal{I}$.
  $S_i \leftarrow S_{i-1} \cup D^+_{best}(u_i) + u_i$.
Return S_i.

OPT can be added to APX, then we might have to discard some elements of OPT. This discard potentially decreases the value of OPT, and therefore, can be seen as the damage incurred by the iteration. Note that by definition of a k-extendible system, we do not have to discard more than k elements for every element we add. That is, at every iteration, only up to $k(\mathcal{D}^+_F + 1)$ elements must be discarded. Therefore, if we manage to upper bound the damage of discarding a single element by the benefit of the allocation at the same iteration, we get the desired bound.[7] Recall that the supermodular dependencies of an element are exactly the elements that may increase its marginal value. Therefore, when discarding an element from OPT, the maximum damage is bounded by the marginal value of this element with respect to its supermodular dependencies in OPT. But, as any subset of OPT can be added to APX, the greedy choice of Algorithm 1 explicitly takes into account the possibility of adding this element and its supermodular dependencies to APX. If another option is chosen, it must have at least the same immediate benefit, as wanted.

We now give a formal proof for Theorem 12. Let us begin with the following observation.

▶ **Observation 17.** *Whenever S_i is not a base, there exists an element $u \in \mathcal{N} \setminus S_i$ for which $S_i \cup \varnothing + u \in \mathcal{I}$ (note that $\varnothing \subseteq \mathcal{D}^+(u)$). Hence, Algorithm 1 always outputs a base.*

Throughout this section, we denote $d = \mathcal{D}^+_f$. Our proof is by a hybrid argument. That is, we have a sequence of hybrid solutions, one per iteration, where the first hybrid contains an optimal solution (and hence, has an optimal value), and the last hybrid is our approximate solution.[8] Roughly speaking, we show the following:

1. By adding each element to the approximate solution, we do not lose more than k elements of the iteration's hybrid (note that we add to our solution at most $d + 1$ elements at any given iteration). This is formalized in Lemma 18, and the proof is based on Definition 3 (k-extendible system).
2. The damage from losing an element of an iteration's hybrid is bounded by the profit the algorithm gains at that iteration. This is formalized in Lemma 19, and the proof is based on Definition 8 (supermodular degree).

In conclusion, we show that when moving from one hybrid to the next, we lose no more than $k(d + 1)$ times the profit at the respective iteration.

[7] The other additive 1 in the denominator of the approximation ratio comes from the fact that by OPT's value we actually mean its marginal contribution to APX. In this sense, addition of elements to APX also might reduce the value of OPT.

[8] Actually, the last hybrid is defined as *containing* our approximate solution, but, as our approximate solution is a base, the hybrid must be exactly equal to it.

Let us formalize the above argument. Let ℓ be the number of iterations performed by Algorithm 1, *i. e.*, ℓ is the final value of i. We recursively define a series of $\ell + 1$ hybrid solutions as follows.

- H_0 is a base containing OPT. By monotonicity, $f(H_0) = f(OPT)$.
- For every $1 \leq i \leq \ell$, H_i is a maximum size independent subset of $H_{i-1} \cup S_i$ containing S_i.

▶ **Lemma 18.** *For every iteration* $1 \leq i \leq \ell$, $|H_{i-1} \setminus H_i| \leq k \cdot |S_i \setminus H_{i-1}| \leq k(d+1)$.

Proof. Let us denote the elements of $S_i \setminus H_{i-1}$ by $v_1, v_2, \ldots, v_r$. We prove by induction that there exists a collection of sets $Y_1, Y_2, \ldots, Y_r$, each of size at most k, such that: $Y_j \subseteq H_{i-1} \setminus (S_{i-1} \cup \{v_h\}_{h=1}^{j-1})$ and $H_{i-1} \setminus (\cup_{h=1}^{j} Y_h) \cup \{v_h\}_{h=1}^{j} \in \mathcal{I}$ for every $0 \leq j \leq r$. For ease of notation, let us denote $Y_1^j = \cup_{h=1}^{j} Y_h$ and $v_1^j = \{v_h\}_{h=1}^{j}$. Using this notation, the claim we want to prove can be rephrased as follows: there exists a collection of sets $Y_1, Y_2, \ldots, Y_r$, each of size at most k, such that: $Y_j \subseteq H_{i-1} \setminus (S_{i-1} \cup v_1^{j-1})$ and $(H_{i-1} \setminus Y_1^j) \cup v_1^j \in \mathcal{I}$ for every $0 \leq j \leq r$.

For $j = 0$ the claim is trivial since $H_{i-1} \in \mathcal{I}$. Thus, let us prove the claim for j assuming it holds for $j-1$. By the induction hypothesis, $(H_{i-1} \setminus Y_1^{j-1}) \cup v_1^{j-1} \in \mathcal{I}$. On the other hand, $S_{i-1} \cup v_1^{j-1}$ is a subset of this set which is independent even if we add v_j to it. Since $(\mathcal{N}, \mathcal{I})$ is a k-extendible system, this implies the existence of a set Y_j of size at most k such that:

$$Y_j \subseteq [(H_{i-1} \setminus Y_1^{j-1}) \cup v_1^{j-1}] \setminus [S_{i-1} \cup v_1^{j-1}] \subseteq H_{i-1} \setminus (S_{i-1} \cup v_1^{j-1}) \ ,$$

and:

$$[(H_{i-1} \setminus Y_1^{j-1}) \cup v_1^{j-1}] \setminus Y_j + v_j \in \mathcal{I} \Rightarrow (H_{i-1} \setminus Y_1^{j}) \cup v_1^{j} \in \mathcal{I} \ ,$$

which completes the induction step. Thus, $(H_{i-1} \setminus Y_1^r) \cup v_1^r \in \mathcal{I}$ is a subset of $H_{i-1} \cup S_i$ which contains S_i and has a size of at least: $|H_{i-1}| - rk + r$. On the other hand, H_i is a maximum size independent subset of $H_{i-1} \cup S_i$, and thus: $|H_i| \geq |H_{i-1}| - rk + r$. Finally, all the elements of H_i belong also to H_{i-1} except, maybe, the elements of $S_i \setminus S_{i-1}$. Hence,

$$|H_{i-1} \setminus H_i| \leq |H_{i-1}| - |H_i| + |S_i \setminus S_{i-1}| \leq |H_{i-1}| - (|H_{i-1}| - rk + r) + r = rk \ .$$

Lemma 18 now follows, since $r \leq d + 1$. ◀

The following lemma upper bounds the loss of moving from one hybrid to the next one.

▶ **Lemma 19.** *For every iteration* $1 \leq i \leq \ell$, $f(H_{i-1}) - f(H_i) \leq k(d+1) \cdot f(D^+_{best}(u_i) + u_i \mid S_{i-1})$, *where* u_i *and* $D^+_{best}(u_i)$ *are the greedy choices made by Algorithm 1 at iteration* i.

Proof. Order the elements of $H_{i-1} \setminus H_i$ in an arbitrary order $v_1, v_2, \ldots v_r$, and let $\bar{H}_j = H_{i-1} \setminus \{v_h \mid 1 \leq h \leq j\}$. For every $1 \leq j \leq r$,

$$\begin{aligned} f(\mathcal{D}^+(v_j) \cap \bar{H}_j + v_j \mid S_{i-1}) &= f(v_j \mid (\mathcal{D}^+(v_j) \cap \bar{H}_j) \cup S_{i-1}) + f(\mathcal{D}^+(v_j) \cap \bar{H}_j \mid S_{i-1}) \\ &\geq f(v_j \mid (\mathcal{D}^+(v_j) \cap \bar{H}_j) \cup S_{i-1}) \geq f(v_j \mid \bar{H}_j \cup S_{i-1}) \ , \end{aligned} \quad (1)$$

where the first inequality follows by monotonicity and the second by Definition 8 (supermodular degree). Specifically, the latter is correct, since the supermodular dependencies of an element are the only ones that can increase its marginal contribution. Therefore, adding

elements of $\bar{H}_j \setminus \mathcal{D}^+(v_j)$ to a set can only decrease the marginal contribution of v_j with respect to this set. Since $\bar{H}_j \cup S_{i-1} = \bar{H}_{j-1} \cup S_{i-1} - v_j$, we get:

$$\sum_{j=1}^{r} f(\mathcal{D}^+(v_j) \cap \bar{H}_j \setminus S_{i-1} + v_j \mid S_{i-1}) = \sum_{j=1}^{r} f(\mathcal{D}^+(v_j) \cap \bar{H}_j + v_j \mid S_{i-1})$$
$$\geq \sum_{j=1}^{r} f(v_j \mid \bar{H}_j \cup S_{i-1}) = f(\bar{H}_0 \cup S_{i-1}) - f(\bar{H}_r \cup S_{i-1}) \geq f(H_{i-1}) - f(H_i) \ ,$$

where the two equalities follow by Definition 6 (marginal set function); the first inequality follows by (1) and the last inequality holds since $\bar{H}_0 = H_{i-1} \supseteq S_{i-1}$ and $\bar{H}_r \cup S_{i-1} \subseteq H_i$. Lemma 19 now follows by recalling that $r \leq k(d+1)$ (by Lemma 18), and noticing that the pair $(v_j, \mathcal{D}^+(v_j) \cap \bar{H}_j \setminus S_{i-1})$ is a candidate pair that Algorithm 1 can choose on Line 4 for every element $v_j \in H_{i-1} \setminus H_i$. ◀

▶ **Corollary 20.** *Algorithm 1 is a $1/(k(d+1)+1)$-approximation algorithm.*

Proof. Adding up Lemma 19 over $1 \leq i \leq \ell$, we get:

$$k(d+1) \cdot [f(S_\ell) - f(S_0)] = k(d+1) \cdot \sum_{i=1}^{\ell} f(D^+_{best}(u_i) + u_i \mid S_{i-1})$$
$$\geq \sum_{i=1}^{\ell} [f(H_{i-1}) - f(H_i)] = f(H_0) - f(H_\ell) \ .$$

Notice that $H_\ell = S_\ell$ because S_ℓ is a base, and therefore, every independent set containing S_ℓ must be S_ℓ itself. Recall also that $f(H_0) = f(OPT)$ and $f(S_0) \geq 0$. Plugging these observations into the previous inequality gives:

$$k(d+1) \cdot f(S_\ell) \geq f(OPT) - f(S_\ell) \Rightarrow f(S_\ell) \geq \frac{f(OPT)}{k(d+1)+1} \ .$$ ◀

3.1.1 A Tight Example for Algorithm 1

In this section we present an example showing that our analysis of Algorithm 1 is tight even when the independence system $(\mathcal{N}, \mathcal{I})$ belongs to k-intersection (recall that any independence system that is k-intersection is also k-extendible, but not vice versa).

▶ **Theorem 21.** *For every $k \geq 1$, $d \geq 0$ and $\varepsilon > 0$, there exists a k-intersection independence system $(\mathcal{N}, \mathcal{I})$ and a function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ with $\mathcal{D}^+_f = d$ for which Algorithm 1 produces a $(1+\varepsilon)/(k(d+1)+1)$ approximation.*

The rest of this section is devoted for constructing the independence system guaranteed by Theorem 21. Let $\mathcal{T}$ be the collection of all sets $T \subseteq \{1, 2, \ldots, k+1\} \times \{0, 1, \ldots, (d+1)(k+1)-1\}$ obeying the following properties:

- For every $1 \leq i \leq k+1$, there exists exactly one x such that T contains the pair (i, x).
- At least one pair (i, x) in T has $x \leq d$.
- Let x_{k+1} be such that $(k+1, x) \in T$. Then $x_{k+1} = 0$ or $x_{k+1} > d$.

Intuitively, the first requirement means that we can view a set $T \in \mathcal{T}$ as a point in a $(k+1)$-dimensional space. The other two requirements make some points illegal. For $k = 1$, the space is a $2(d+1) \times 2(d+1)$ grid, and the legal points are the ones that are either in row 0 or in one of the rows $d+1$ to $2(d+1)-1$ and one of the columns 0 to d.

Let $\mathcal{N}$ be the ground set $\{u_T \mid T \in \mathcal{T}\}$. We define k matroids on this ground set as follows. For every $1 \leq i \leq k$, $\mathcal{M}_i = (\mathcal{N}, \mathcal{I}_i)$, where a set $S \subseteq \mathcal{N}$ belongs to $\mathcal{I}_i$ if and only if for every $0 \leq x < (d+1)(k+1)$, $|\{u_T \in S \mid (i,x) \in T\}| \leq 1$. One can easily verify that $\mathcal{M}_i$ is a partition matroid. The independence system we construct is the intersection of these matroids, *i. e.*, it is $(\mathcal{N}, \mathcal{I})$, where $\mathcal{I} = \bigcap_{i=1}^{k} \mathcal{I}_i$. Next, we define the objective function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ as follows.

$$f'(S) = \sum_{x=0}^{(d+1)(k+1)-1} \min\{1, |\{u_T \in S \mid (k+1,x) \in T\}|\} \ .$$

That is, for $k = 1$, f' gains a value of 1 for every row that was "hit" by an element. For every $0 \leq x \leq d$, let $\hat{T}(x) = \{(k+1,0)\} \cup \{(i,x)\}_{i=1}^{k}$ (notice that $\hat{T}_x \in \mathcal{T}$).

$$f(S) = \begin{cases} f'(S) + \varepsilon & \text{if } \{u_{\hat{T}(x)}\}_{x=0}^{d} \subseteq S \ , \\ f'(S) & \text{otherwise} \ . \end{cases}$$

One can check that f' is a non-negative monotone submodular function, and thus, $\mathcal{D}_f^+ = d$.

▶ **Lemma 22.** *Given the above constructed independence system $(\mathcal{N}, \mathcal{I})$ and objective function f, Algorithm 1 outputs a solution of value $1 + \varepsilon$.*

Proof. Consider an arbitrary element $u_T \in \mathcal{N}$. If $T \notin \{\hat{T}(x)\}_{0 \leq x \leq d}$, then $f(u_T \mid S) = f'(u_T \mid S)$ for every set $S \subseteq \mathcal{N}$, and thus, $\mathcal{D}^+(u_T) = \varnothing$ because f' is a submodular function. Hence, for every such u_T we get: $f(\mathcal{D}^+(u_T) + u_T) = 1$. Consider now the case $T \in \{\hat{T}(x)\}_{0 \leq x \leq d}$. In this case, clearly, $\mathcal{D}^+(u_T) = \{u_{\hat{T}(x)}\}_{0 \leq x \leq d} - u_T$, and thus, $f(\mathcal{D}^+(u_T) + u_T) = 1 + \varepsilon$. In conclusion, Algorithm 1 picks exactly the elements of $\{u_{\hat{T}(x)}\}_{0 \leq x \leq d}$ to its solution in the first iteration.

To complete the proof we still need to show that Algorithm 1 cannot increase the value of its solution in the next iterations. Consider an arbitrary element $u_T \in \mathcal{N} \setminus \{u_{\hat{T}(x)}\}_{0 \leq x \leq d}$. By definition, T must contain a pair (i,x) such that $0 \leq x \leq d$. There are two cases:

- If $i \neq k+1$, then u_T cannot coexist in an independent set of $\mathcal{M}_i$ with $u_{\hat{T}(x)}$ because both correspond to sets containing the pair (i,x).
- If $i = k+1$, then $x = 0$ because $u_T \in \mathcal{N}$.

From the above analysis, we get that all elements added to the solution after the first iteration contain the pair $(k+1,0)$ (and thus, no other pair of the form $(k+1,x)$). Hence, they do not increase the value of either f' or f. ◀

To prove Theorem 21, we still need to show that $(\mathcal{N}, \mathcal{I})$ contains an independent set of a high value. Consider the set $S = \{u_{\bar{T}(j)}\}_{j=0}^{k(d+1)}$, where $\bar{T}(j) = \{(i,x) \mid 1 \leq i \leq k+1 \text{ and } x = (i(d+1) - j) \bmod (d+1)(k+1)\}$.

▶ **Observation 23.** *$S \subseteq \mathcal{N}$.*

Proof. We need to show that for every $0 \leq j \leq k(d+1)$, $u_{\bar{T}(j)} \in \mathcal{N}$. For $j = 0$, the only pair of the form $(k+1,x)$ in $\bar{T}(0)$ is $(k+1,0)$, which completes the proof. Thus, we may assume from now on $1 \leq j \leq k(d+1)$, and let $i = \lceil j/(d+1) \rceil$. For shortness, let us denote $h = d+1$. Clearly $1 \leq i \leq k$ and $\bar{T}(j)$ contains the pair (i,x) for:

$$x = (ih - j) \bmod h(k+1) = (\lceil j/h \rceil \cdot h - j) \bmod h(k+1) \ .$$

To prove the observation, we need to show that $0 \leq x \leq d$. This follows since $\lceil j/h \rceil \cdot h - j \geq (j/h) \cdot h - j = 0$ and $\lceil j/h \rceil \cdot h - j < \lfloor j/h + 1 \rfloor \cdot h - j = h = d+1$. ◀

▶ **Observation 24.** *For every two values* $0 \leq j_1 < j_2 \leq k(d+1)$, $\bar{T}(j_1) \cap \bar{T}(j_2) = \varnothing$. *Hence* $S \in \mathcal{I}$ *and* $f(S) \geq f'(S) = |S| = k(d+1)+1$.

Proof. Assume for the sake of contradiction that $(i,x) \in \bar{T}(j_1) \cap \bar{T}(j_2)$. Then, modulo $(d+1)(k+1)$, the following equivalence must hold:

$$(i(d+1) - j_1) \equiv (i(d+1) - j_2) \Rightarrow j_1 \equiv j_2 \ ,$$

which is a contradiction since $j_1 \neq j_2$ and they are both in the range $[0, k(d+1)]$. ◀

3.2 Hardness (Proof of Theorem 14)

Before proving Theorem 14 let us state the hardness result of [22] given by Theorem 25. In the r-Dimensional Matching problem one is given an r-sided hypergraph $G = (\bigcup_{i=1}^{r} V_i, E)$, where every edge $e \in E$ contains exactly one vertex of each set V_i. The objective is to select a maximum size matching $M \subseteq E$, *i. e.*, a subset $M \subseteq E$ of edges which are pairwise disjoint.

▶ **Theorem 25** (Hazan et al. [22])**.** *It is NP-hard to approximate* r*-Dimensional Matching to within* $O(\log r/r)$ *in polynomial time, even if* r *is a constant.*

Theorem 14 follows by combining Theorem 25 with the following lemma.

▶ **Lemma 26.** *Any instance of* r*-Dimensional Matching can be represented as maximizing a monotone function* f *with* $\mathcal{D}_f^+ = \mathcal{D}_f \leq d$ *over a* k*-intersection set system for every* $d \geq 0$ *and* $k \geq 1$ *obeying* $r \leq k(d+1)$.

Proof. For simplicity, assume $r = k(d+1)$. Let $G = (\bigcup_{i=1}^{r} V_i, E)$ be the graph representing the r-Dimensional Matching instance. We first construct a new graph G' as follows. For every edge $e \in E$ and $1 \leq j \leq d+1$, let $e(j) = e \cap (\bigcup_{i=(j-1)k+1}^{jk} V_i)$, *i. e.*, $e(j)$ is the part of e hitting the vertex sets $V_{(j-1)k+1}, \ldots, V_{jk}$. The edges of the new graph $G' = (\bigcup_{i=1}^{r} V_i, E')$ are then defined as all the edges that can be obtained this way. More formally:

$$E' = \{e(j) \mid e \in E \text{ and } 1 \leq j \leq d+1\} \ .$$

It is easy to see that the original instance of r-Dimensional Matching is equivalent to the problem of finding a matching in G' maximizing the objective function $f : 2^{E'} \to \mathbb{R}^+$ defined as follows.

$$f(S) = \sum_{e \in E} \left\lfloor \frac{|S \cap \{e(j) \mid 1 \leq j \leq d+1\}|}{d+1} \right\rfloor \ .$$

Moreover, $\mathcal{D}_f = \mathcal{D}_f^+ = d$. Thus, to complete the proof we only need to show that the set of all legal matchings of G' can be represented as a k-intersection independence system.

Consider the following partition of the vertices of G'. For every $1 \leq j \leq k$, $V'_j = \bigcup_{i=0}^{d} V_{j+ki}$. Observe that each edge of G' contains exactly one vertex of V'_j. Hence, the constraint that no two edges intersect on a node of V'_j can be represented by the partition matroid $M_j = (E', \mathcal{I}_j)$ defined as following. A set $S \subseteq E'$ is in $\mathcal{I}_j$ if and only if no two edges of S intersect on a node of V'_j. The set of legal matchings of G' is, then, exactly $\bigcap_{j=1}^{k} \mathcal{I}_j$. ◀

4 Uniform Matroid Constraint

In this section we prove Theorems 15 and 16.

4.1 Algorithm for Uniform Matroid Constraint (Proof of Theorem 15)

Algorithm 1 given in Section 3 provides a $1/(\mathcal{D}_f^+ + 2)$ approximation for a general k-extendible constraint. In this section, our objective is to improve over this approximation ratio for uniform matroid constraints. Throughout this section we use d to denote $\mathcal{D}_f^+$ and OPT to denote an arbitrary optimal solution.

A natural greedy approach (similar to Algorithm 1) is to repeatedly select the set $\mathcal{D}^+(u) + u$ with the maximal marginal contribution with respect to the current solution, and add it to the solution. When $d+1$ divides k, this works well, and can be shown to have an approximation ratio of $1 - e^{1/(d+1)}$. However, when $d+1$ does not divide k, the algorithm might not be able to perform $\lceil k/(d+1) \rceil$ iterations, which results in an inferior approximation ratio.

A possible workaround is to add a final iteration in which the algorithm can select any subset of $\mathcal{D}^+(u) + u$ (for any $u \in \mathcal{N}$). This allows the algorithm to perform $\lceil k/(d+1) \rceil$ iterations on all instances, and it is an open problem to analyse its approximation ratio.

An alternative workaround is to reduce into the case of k divisible by $d+1$. This is the idea behind Algorithm 2. Observe that $d'+1$ divides $k-r$ (the size of the part of OPT we do not guess). When we say that the algorithm "guesses" values, we simply mean "sequentially tries all possible values".

Algorithm 2: Guess Greedy(f, k)

1 Guess: $d' = \max_{u \in OPT} |\mathcal{D}^+(u) \cap OPT|$, an element u^* for which $|\mathcal{D}^+(u^*) \cap OPT| = d'$ and the set $C = \mathcal{D}^+(u^*) \cap OPT$ itself.
2 Initialize: $r = k \bmod (d'+1)$, $\ell = (k-r)/(d'+1)$ and $S_0 \subseteq C$ as an arbitrary subset of size r.
3 **for** $i = 1$ **to** ℓ **do**
4 Let $u_i \in \mathcal{N}$ and $D^+_{best}(u_i) \subseteq \mathcal{D}^+(u_i)$ be a pair of an element and a set of size at most d' maximizing $f(D^+_{best}(u_i) + u_i \mid S_{i-1})$.
5 $S_i \leftarrow S_{i-1} \cup D^+_{best}(u_i) + u_i$.
6 Return S_ℓ.

▶ **Observation 27.** *Algorithm 2 returns a feasible solution and has a polynomial time complexity in n and 2^d.*

Due to space constraints, we only sketch the analysis of the approximation ratio of Algorithm 2. To simplify notation, define $k' \stackrel{\text{def}}{=} k - r$. Observe that $\ell = k'/(d'+1)$.

▶ **Lemma 28.** *For every $0 \leq i \leq \ell$, $f(S_i) \geq (1 - 1/k')^i \cdot f(S_0) + [1 - (1 - 1/k')^i] \cdot f(OPT)$.*

Proof Sketch. We prove the lemma by induction. For $i = 0$ the claim is trivial since $f(S_0) = (1-1/k')^0 \cdot f(S_0) + [1 - (1-1/k')^0] \cdot f(OPT)$. Next, assume the claim holds for $i-1$, and let us prove it for $i > 0$. Observe that $S_0 \subseteq OPT$, and therefore, $|OPT \setminus S_0| = k'$. Using the same methods used in the proof of Lemma 19, it can be shown that:

$$\sum_{u \in OPT \setminus S_0} f(\mathcal{D}^+(u) \cap OPT + u \mid S_{i-1}) \geq f(OPT) - f(S_{i-1}) \ .$$

Notice that the pair $(u, \mathcal{D}^+(u) \cap OPT)$ is a candidate pair to be selected as $(u_i, D^+_{best}(u_i))$ for every element $u \in OPT \setminus S_0$. Hence, $f(D^+_{best}(u_i) + u_i \mid S_{i-1}) \geq [f(OPT) - f(S_{i-1})]/k'$.

Thus,

$$f(S_i) \geq [f(OPT) - f(S_{i-1})]/k' + f(S_{i-1}) = f(OPT)/k' + (1 - 1/k') \cdot f(S_{i-1})$$
$$\geq (1 - 1/k')^i \cdot f(S_0) + [1 - (1 - 1/k')^i] \cdot f(OPT) \ ,$$

where the last inequality follows by induction hypothesis. ◀

▶ **Corollary 29.** *The approximation ratio of Algorithm 2 is* $1 - e^{-1/(d'+1)} \geq 1 - e^{-1/(d+1)}$.

4.2 Hardness (Proof of Theorem 16)

The Gap Small-Set Expansion problem (introduced by [31]) is the following promise problem.

▶ **Problem 1** (Gap Small-Set Expansion(η, δ))**.**
Input: An undirected graph $G = (V, E)$.
Output: Is $\phi_G(\delta) \geq 1 - \eta$ *or* $\phi_G(\delta) \leq \eta$*? (*$\phi_G(\delta)$ *is the edge expansion of* G *with respect to subsets of size exactly* $\delta|V|$*.)*

The Small-Set Expansion Hypothesis (SSE), introduced by Raghavendra and Steurer [31] (see, also, [32]) is the following.

▶ **Hypothesis 1.** *For every* $\eta > 0$*, there exists* δ *such that Problem 1 parametrized by* η *and* δ *is* $\mathcal{NP}$*-hard.*

A hypergraph representation $F = (V_F, E_F, w_f)$ of a set function $f : 2^{\mathcal{N}} \to \mathbb{R}^+$ (defined by [5, 7]) is the following. The set V_F contains exactly a single vertex for each element of the ground set $\mathcal{N}$. The set E_F is the set of the hyperedges of the hypergraph. The function $w_F : E_F \to \mathbb{R}$ assigns a real value for each hyperedge $e \in E_F$, and these values obey the following property: for every set $S \subseteq \mathcal{N}$, the sum of the values of the hyperedges in the hypergraph induced by the vertices representing S is exactly $f(S)$. It is well known that any set function (normalized to have $f(\varnothing) = 0$) can be uniquely represented by a hypergraph representation and vice versa.

Using the above definition, we show that maximizing a monotone set function subject to a uniform matroid constraint captures Problem 1. Let $G = (V, E)$ be an arbitrary instance of Problem 1 with parameters η and δ. We construct from G an hypergraph representation $F = (V_F, E_F, w_F)$. The sets V_F and E_F are chosen as identical to V and E, respectively (*i. e.*, all the hyperedges are of rank 2, hence, the hypergraph is in fact a graph). The value function w_F gives a value of 1 for every edge of E_F. We can now consider the problem of finding a set of size at most $\delta|V|$ maximizing the set function f corresponding to the hypergraph representation F. Theorem 16 follows immediately by the observation that a constant approximation for the last problem implies a constant approximation for Problem 1.

5 Future Research

We view this work as a proof of concept showing that one can obtain interesting results for the problem of maximizing an arbitrary monotone set function subject to non-trivial constraints. We would like to point out two possible directions for future research. The first direction is studying the approximation ratio that can be guaranteed for more general problems as a function of the supermodular degree. Two possible such generalizations are a general k-system constraint and a non-monotone objective. Notice that non-monotone objectives are interesting even in the unconstrained case.

The second direction is determining the guarantees that can be achieved for other complexity measures (with respect to either monotone or non-monotone set functions). Specifically, we would like to draw attention to two complexity measures introduced by [11], namely MPH (for monotone set functions) and PLE (for not necessarily monotone set functions). Both measures are based on fractionally sub-additive functions, a strict super-class of submodular functions, and they generally give lower values to set functions in comparison to the supermodular degree. Thus, it is intriguing to show positive results for either of these measures.

Acknowledgments. We are grateful to Uri Feige and Irit Dinur for valuable discussions.

References

1 Sushil Bikhchandani and John W. Mamer. Competitive equilibrium in an exchange economy with indivisibilities. *Journal of Economic Theory*, 74(2):385–413, 1997.

2 Liad Blumrosen and Noam Nisan. On the computational power of demand queries. *SIAM Journal on Computing*, 39:1372–1391, 2009.

3 Niv Buchbinder, Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. In *FOCS*, pages 649–658, 2012.

4 Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

5 Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet. Multiagent resource allocation in k-additive domains: preference representation and complexity. *Annals of Operations Research*, 163:49–62, 2008.

6 M. Conforti and G. Cornuèjols. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Disc. Appl. Math.*, 7(3):251–274, 1984.

7 V. Conitzer, T. Sandholm, and P. Santi. Combinatorial auctions with k-wise dependent valuations. In *AAAI*, pages 248–254, 2005.

8 Shahar Dobzinski, Noam Nisan, and Michael Schapira. Approximation algorithms for combinatorial auctions with complement-free bidders. In *STOC*, pages 610–618, New York, NY, USA, 2005. ACM.

9 Shahar Dobzinski and Michael Schapira. An improved approximation algorithm for combinatorial auctions with submodular bidders. In *SODA*, pages 1064–1073, 2006.

10 Uriel Feige. On maximizing welfare when utility functions are subadditive. *SIAM Journal on Computing*, 39:122–142, 2009. Preliminary version in STOC'06.

11 Uriel Feige, Michal Feldman, Nicole Immorlica, Rani Izsak, Brendan Lucier, and Vasilis Syrgkanis. A unifying hierarchy of valuations with complements and substitutes, 2014. Working paper.

12 Uriel Feige and Rani Izsak. Welfare maximization and the supermodular degree. In *ITCS*, pages 247–256, 2013.

13 Uriel Feige and Jan Vondrák. The submodular welfare problem with demand queries. *Theory of Computing*, 6(1):247–290, 2010.

14 Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz. A unified continuous greedy algorithm for submodular maximization. In *FOCS*, 2011.

15 Moran Feldman, Joseph (Seffi) Naor, Roy Schwartz, and Justin Ward. Improved approximations for k-exchange systems. In *ESA*, pages 784–798, 2011.

16 M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions – II. In *Polyhedral Combinatorics*, volume 8 of *Mathematical Programming Study*, pages 73–87. North-Holland Publishing Company, 1978.

17 Gagan Goel, Chinmay Karande, Pushkar Tripathi, and Lei Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. *SIGecom Exchanges*, 9(1):8, 2010.

18 M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatoria*, 1(2):169–197, 1981.

19 Faruk Gul and Ennio Stacchetti. Walrasian equilibrium with gross substitutes. *Journal of Economic Theory*, 87(1):95–124, 1999.

20 D. Hausmann and B. Korte. K-greedy algorithms for independence systems. *Oper. Res. Ser. A-B*, 22(1):219–228, 1978.

21 D. Hausmann, B. Korte, and T. Jenkyns. Worst case analysis of greedy type algorithms for independence systems. *Math. Prog. Study*, 12:120–131, 1980.

22 Elad Hazan, Shmuel Safra, and Oded Schwartz. On the complexity of approximating k-set packing. *Computational Complexity*, 15(1):20–39, May 2006.

23 Satoru Iwata and Kiyohito Nagano. Submodular function minimization under covering constraints. In *FOCS*, pages 671–680, 2009.

24 Satoru Iwata and James B. Orlin. A simple combinatorial algorithm for submodular function minimization. In *SODA*, pages 1230–1237, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

25 T. Jenkyns. The efficacy of the greedy algorithm. *Cong. Num.*, 17:341–350, 1976.

26 B. Korte and D. Hausmann. An analysis of the greedy heuristic for independence systems. *Annals of Discrete Math.*, 2:65–74, 1978.

27 Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Math. Oper. Res.*, 35(4):795–806, 2010.

28 Julián Mestre. Greedy in approximation algorithms. In *ESA*, pages 528–539, 2006.

29 G. Nemhauser and L. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Math. Oper. Res.*, 3(3):177–188, 1978.

30 G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions – I. *Mathematical Programming*, 14:265–294, 1978.

31 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *STOC*, pages 755–764, 2010.

32 Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *IEEE Conference on Computational Complexity*, pages 64–73, 2012.

33 Jan Vondrák. Symmetry and approximability of submodular maximization problems. *SIAM J. Comput.*, 42(1):265–304, 2013.

34 Justin Ward. A (k+3)/2-approximation algorithm for monotone submodular k-set packing and general k-exchange systems. In *STACS*, pages 42–53, 2012.

On the Equivalence of the Bidirected and Hypergraphic Relaxations for Steiner Tree

Andreas Emil Feldmann[1], Jochen Könemann[1], Neil Olver[2], and Laura Sanità[1]

1 **Department of Combinatorics and Optimization, University of Waterloo**
{andreas.feldmann,jochen,laura.sanita}@uwaterloo.ca
2 **VU University & CWI, Amsterdam**
n.olver@vu.nl

Abstract

The bottleneck of the currently best $(\ln(4)+\varepsilon)$-approximation algorithm for the NP-hard *Steiner tree* problem is the solution of its large, so called *hypergraphic*, linear programming relaxation (HYP). Hypergraphic LPs are NP-hard to solve exactly, and it is a formidable computational task to even approximate them sufficiently well.

We focus on another well-studied but poorly understood LP relaxation of the problem: the *bidirected cut relaxation* (BCR). This LP is compact, and can therefore be solved efficiently. Its integrality gap is known to be greater than 1.16, and while this is widely conjectured to be close to the real answer, only a (trivial) upper bound of 2 is known.

In this paper, we give an efficient constructive proof that BCR and HYP are polyhedrally equivalent in instances that do not have an (edge-induced) claw on Steiner vertices, i.e., they do not contain a Steiner vertex with 3 Steiner neighbors. This implies faster $\ln(4)$-approximations for these graphs, and is a significant step forward from the previously known equivalence for (so called *quasi-bipartite*) instances in which Steiner vertices form an independent set. We complement our results by showing that even restricting to instances where Steiner vertices induce one single star, determining whether the two relaxations are equivalent is NP-hard.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Steiner tree, bidirected cut relaxation, hypergraphic relaxation, polyhedral equivalence, approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.176

1 Introduction

In an instance of the well-studied, NP-hard [5, 14] Steiner tree problem one is given an undirected graph $G = (V, E)$, a non-negative cost $\text{cost}(e)$ for each edge $e \in E$, and a set of *terminals* $R \subseteq V$. The goal is to find a minimum-cost tree spanning R. Steiner trees arise in a host of practical applications (e.g., see the survey [12] and the current DIMACS implementation challenge [8]), and therefore have been extensively studied in the network design community.

In this paper, we focus on the problem's efficient approximability. In a recent breakthrough, Byrka et al. [2] presented the currently best $(\ln(4) + \varepsilon)$-approximation algorithm for the problem. The algorithm crucially relies on the repeated solution of a large, so called *hypergraphic* LP relaxation (henceforth abbreviated by HYP) for the problem. It was later shown by Goemans et al. [11] that it is possible to achieve the same approximation guarantee while only solving HYP once. However, solving hypergraphic Steiner tree relaxations

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 176–191

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

is challenging: Goemans et al. [11] also showed that solving them exactly is strongly NP-hard, and even their approximation amounts to (exactly) solving LPs with more than $|R|^k$ variables and constraints (where k is a constant that needs to be ~ 100 in order to yield an approximation to HYP of sufficient quality).

Another well-known formulation for the Steiner tree problem is the *bidirected cut* relaxation (BCR) [6, 20]. BCR is an appealing relaxation as its compactness implies efficient solvability. As one way of obtaining a faster approximation algorithm for the Steiner tree problem, we therefore propose to first compute a solution to HYP *from* a solution to BCR. Then, we apply the algorithm of Goemans et al. [11] in order to compute a Steiner tree with cost at most $\ln(4)$ times that of the given HYP solution. Since HYP has a smaller integrality gap than BCR in general (we present the largest known gap ratio of 12/11 below), we cannot hope to compute a solution to HYP of the same cost as the optimum BCR solution. We therefore ask when these two LPs have the same integrality gap.

The integrality gap of BCR is not well-understood. It is known to be at least $36/31 \approx 1.16$ [2], and while the latter number is widely conjectured to be close to the truth, the only thing known is an almost trivial upper bound of 2. HYP on the other hand has an integrality gap of at least $8/7 \approx 1.14$ [15] and at most $\ln(4) \approx 1.39$ [11]. Hence comparing the gaps of these two LPs can shed some light on the gap of BCR.

Previously it was known that the integrality gaps of BCR and HYP are equal for *quasi-bipartite* instances where no two Steiner vertices (the vertices in $V \setminus R$) are connected by an edge [3, 9, 11]. For these graphs the Steiner tree problem remains NP-hard [18]. In this paper we significantly extend the class of instances where the integrality gaps are identical. In our main result, we show that as long as the input graph G has no Steiner vertex with three Steiner neighbors (we will refer to this as a *Steiner claw*), BCR and HYP are polyhedrally equivalent. Specifically, we will provide a cost-preserving, and efficiently computable map between feasible solutions to BCR and those of HYP. We will also show that our results are nearly best possible by exhibiting instances with a single star on Steiner vertices for which it is NP-hard to decide whether BCR and HYP have the same integrality gap.

In the following we describe the relaxations BCR and HYP in more detail before formally stating our contributions.

1.1 Bidirected and Hypergraphic LPs for Steiner Trees

In the bidirected cut relaxation one usually considers a directed auxiliary graph that has two arcs (u, v) and (v, u) of cost $\text{cost}(uv)$ for each original edge $uv \in E$. The LP, which we will refer to as BCR*, has a variable for each of these arcs, and its constraints force at least one arc to cross each directed cut that separates a chosen root $r \in R$ from at least one other terminal (see [6, 20]). More concretely, if the set $\vec{E}$ contains the directed arcs (u, v) and (v, u) for all edges $uv \in E$, $\delta^+(S) := \{(u, v) \in \vec{E} \mid u \in S, v \notin S\}$ is the set of arcs crossing a set $S \subseteq V$, and $z(\delta^+(S)) = \sum_{a \in \delta^+(S)} z_a$, the LP is

$$\min \quad \sum_{a \in \vec{E}} z_a \, \text{cost}(a) \qquad \text{s. t.} \tag{BCR*}$$

$$z(\delta^+(S)) \geq 1 \qquad \forall S \subseteq V \setminus \{r\}, S \cap R \neq \emptyset$$

$$z \geq \mathbb{0}$$

In this paper, we importantly choose to work with an equivalent *undirected* formulation (see [10]) which we will refer to as BCR. We state this LP below, where we associate a variable z_e with each (undirected) edge $e \in E$, and a variable y_v with each vertex $v \in V$. For

brevity we use $E(S)$ for the collection of edges with both ends in $S \subseteq V$, $z(E') = \sum_{e \in E'} z_e$, $y(S) = \sum_{v \in S} y_v$, and $y_{max}(S)$ as a shorthand for $\max_{v \in S} y_v$.

$$\min \quad \sum_{e \in E} z_e \operatorname{cost}(e) \qquad \text{s. t.} \tag{BCR}$$

$$\begin{aligned} z(E(S)) &\leq y(S) - y_{max}(S) && \forall S \subseteq V \\ z(E) &= y(V) - 1 \\ y_t &= 1 && \forall t \in R \\ y, z &\geq \mathbb{0} \end{aligned}$$

We note that the LP becomes Edmonds' famous *subtour* formulation for the spanning tree polyhedron [7] when y is replaced by the vector of ones. Furthermore, BCR can be solved efficiently: simply compute a solution to a compact flow formulation of its directed counterpart BCR*, and observe that it can be mapped to a solution of the same value for BCR (see [10]): set y_v in BCR to the sum of outgoing arc values from $v \in V \setminus \{r\}$ in BCR* (this corresponds to the amount of flow that v can send to the root). The value y_r of the root of BCR* is simply 1 in BCR. The value z_e for an edge in BCR is given by the sum of the corresponding two arc values in BCR*.

Hypergraphic LPs are inspired by the observation that the Steiner tree problem can be equivalently phrased as that of computing a minimum-cost spanning tree in an appropriately defined hypergraph on the terminals. There are multiple equivalent, directed and undirected forms of HYP [3]. Corresponding to our undirected choice of BCR, we will henceforth focus on the hypergraphic subtour relaxation introduced in [19]. The LP has one variable for each *full-component* of the instance. A full-component is a tree all of whose leaves are terminals, and whose internal nodes are Steiner vertices. We let $\mathcal{K}$ be the set of all full-components of the instance, and note that $\mathcal{K}$ may have multiple full-components spanning the same set of terminals, but having different edges. The cost of a full-component is equal to the sum of the cost of its edges. In the following hypergraphic subtour formulation we let $(a)^+$ be a short-hand for $\max\{0, a\}$, and $R(C)$ denote the set of terminals included in C .

$$\min \quad \sum_{C \in \mathcal{K}} x_C \operatorname{cost}(C) \qquad \text{s. t.} \tag{HYP}$$

$$\begin{aligned} \sum_{C \in \mathcal{K}} x_C (|R(C) \cap S| - 1)^+ &\leq |S| - 1 && \forall S \subseteq R, S \neq \emptyset \\ \sum_{C \in \mathcal{K}} x_C (|R(C)| - 1)^+ &= |R| - 1 \\ x &\geq \mathbb{0} \end{aligned}$$

As mentioned, solving HYP is strongly NP-hard. However, restricting $\mathcal{K}$ to full-components spanning at most k terminals (for some fixed k) renders the LP polynomial-time solvable, and it can be shown that its optimal value increases by at most a factor of $(1 + 1/\lfloor \log k \rfloor)$ [1]. We may therefore choose $k = k(\varepsilon)$ appropriately to obtain a $1 + \varepsilon$ approximation to HYP, for any $\varepsilon > 0$. As mentioned above, to achieve solutions of sufficient quality, k needs to be ~ 100, which implies LPs with more than $|R|^{100}$ variables and constraints.

1.2 Our Contributions

We call a Steiner tree instance *Steiner claw-free* if the graph G has no Steiner vertex with at least three Steiner neighbors. Our main result is the following, which implies

faster ln(4)-approximations for Steiner claw-free graphs. In particular, our running time is dominated by solving BCR, which in its compact flow formulation has $O(|R||E|)$ variables and constraints.

▶ **Theorem 1.** *In a Steiner claw-free Steiner tree instance, any minimal solution to BCR can be efficiently converted to a solution to HYP of no larger cost.*

As an immediate consequence, we obtain an integrality gap bound of ln(4) for BCR in Steiner claw-free instances via [11], improving the previously known bound of 2. The only class of Steiner tree instances where BCR was previously known to exhibit an integrality gap smaller than 2 is that of quasi-bipartite graphs. Previous work in [4, 11] showed that their integrality gap is at most $73/60 \approx 1.216$.

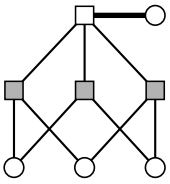


Figure 1 Example instance with HYP $\neq$ BCR. Terminals are circles, Steiner vertices are squares.

Theorem 1 implies that BCR and HYP are equivalent in every instance of the Steiner tree problem where Steiner vertices induce subgraphs in which the maximum degree of each vertex is 2 (i.e. paths and cycles). On the other hand, Figure 1 shows a graph with 4 Steiner vertices inducing a subgraph with only one vertex of degree 3, where BCR and HYP are not equivalent. If we assume all edges to have unit cost, BCR is easily seen to admit a solution of cost 5.5: let $z_e = 1$ for the thick edge, and $z_e = 1/2$ otherwise. All white vertices v in the figure have $y_v = 1$, and others have $y_v = 1/2$. The optimum Steiner tree has cost 6 and this is also the value of HYP. Hence in general the gap ratio between the two LPs is at least 12/11.

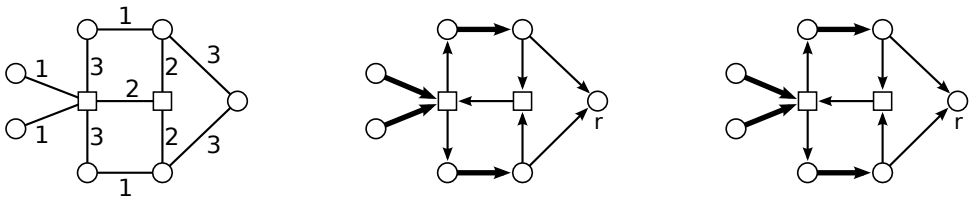
At a high level, our algorithmic proof of Theorem 1 follows the greedy approach taken in [9, 11] for quasi-bipartite instances. Roughly, the above papers first solve the directed version BCR* of BCR, and convert it into a solution for a directed version of HYP, commonly referred to as *directed component relaxation* DCR (see [17]). This directed formulation is equivalent to HYP [3]. For DCR, each full-component is directed, i.e. it is an in-arborescence to one of its terminals, called its *head*. We call the set of all directed full-components $\vec{\mathcal{K}}$. By $\Delta^+(S)$ we denote all full-components $C \in \vec{\mathcal{K}}$ for which the head lies outside S, while some other terminal of C lies inside. Also let $x(\Delta^+(S)) = \sum_{C \in \Delta^+(S)} x_C$. The directed hypergraphic (component) relaxation then is:

$$\min \quad \sum_{C \in \vec{\mathcal{K}}} x_C \, \mathrm{cost}(C) \quad \text{s.t.} \qquad \text{(DCR)}$$

$$x(\Delta^+(S)) \geq 1 \qquad \forall S \subseteq R \setminus \{r\}, S \neq \emptyset$$

$$x \geq \mathbb{0}$$

The approach of [9, 11] is to iteratively and greedily *shave off* fractional capacity uniformly from the arcs of a directed full-component in the support of the given directed BCR* solution. In the case of quasi-bipartite instances, this approach works and yields a feasible solution for DCR of the same cost as the original BCR* solution. As soon as Steiner vertices are allowed to have Steiner neighbors, the above strategy runs into problems, however. Figure 2(a) shows a Steiner claw-free instance, and two optimal solutions to BCR* in Figures 2(b) and 2(c). One can show that there is no DCR solution whose canonical *projection* yields (or more precisely, is dominated by) the BCR* solution in 2(c). Hence the outlined greedy strategy taken in [9, 11] will not work here. On the other hand, the solution given in 2(b) is the projection of a feasible solution to DCR. The crux appears to be that both solutions in Figure 2(b) and 2(c) *project* to the same undirected solutions of BCR. By considering undirected relaxations we avoid the complication inherent in the directed nature of the LPs.



(a) Edge costs.

(b) Decomposable: one full-component is marked a, the other b, each with value 1/2.

(c) Not decomposable.

Figure 2 An instance with edge costs as given in (a). Some optimal solutions to BCR* are decomposable (b) into a DCR solution, and others (c) are not (we omit the proof). In the BCR* solutions the root is marked r, bold arcs have capacity 1, and the others 1/2.

The results for the quasi-bipartite case [3, 9, 11] at their heart rely on the property that tight sets that intersect in terminals can be uncrossed. To move beyond the quasi-bipartite case, however, we require a deeper understanding of the interaction of tight sets, including those that are not terminal-intersecting. We believe that our techniques may be helpful in the quest for a better-than-2 bound on the integrality gap of the bidirected cut relaxation: while a mapping from BCR to HYP that preserves cost is not possible, one that only loses a small factor may be. In fact, it can readily be seen that the algorithm we present in the rest of this paper can be used to compute an approximate solution to HYP for the example given in Figure 1. For this we set $y_v = 1$ for any one of the gray Steiner vertices v, and also $z_e = 1$ for the edge e connecting v to the white Steiner vertex. This again yields a feasible solution to BCR for which our algorithm computes a solution to HYP of the same cost 6.

Our main result of this paper shows that the property of being Steiner claw-free, which is polynomially checkable, is a sufficient condition for equivalence of the two relaxations. We also show that there is no good characterization of this equivalence. Even if we restrict to instances where the Steiner vertices induce a single star, deciding equivalence is NP-hard. We defer the proof of the following theorem to the full version of the paper.

▶ **Theorem 2.** *It is NP-hard to decide for a given Steiner tree instance whether BCR has the same optimum value as HYP, even if the Steiner vertices induce a single star.*

2 A Constructive Map Between BCR and HYP

In this section, we will give a detailed description of an algorithm that converts a minimal feasible BCR solution into a solution for HYP. At a high level, the arguments are structured similarly to those used in [9, 11]. Crucially, however, we will be using the undirected relaxations BCR and HYP introduced in the previous section instead of their directed analogs.

Our algorithm computes a solution to BCR and in each step identifies a tree C^* in the support of this solution. We then carefully choose $\varepsilon > 0$, and remove it from z_e for all $e \in E(C^*)$ as well as from y_v for all $v \in V(C^*) \setminus R$. Subsequently, we then add ε to the x-variable of a maximal full-component contained in C^*. In order to facilitate our discussion of this greedy process, we define the following "mixed LP" (M), which is a hybrid of BCR and HYP.

Algorithm 1 Finding a full component.

1: Choose an arbitrary Steiner vertex $\ell \in V(H)$ and let $V(C^*) = \{\ell\}$.
2: As long as there is a Steiner vertex v neighboring a vertex $w \in V(C^*)$, add it and the edge vw to C^* as long as $C^*[S]$ is connected for all tight sets S.
3: As long as there is a terminal t neighboring a Steiner vertex $w \in V(C^*)$, add it and the edge wt to C^* as long as $C^*[S]$ is connected for all tight sets S.
4: Obtain C from C^* by deleting Steiner leaves as long as these exist.

$$\min \quad \sum_{e\in E} z_e \operatorname{cost}(e) \; + \; \sum_{C\in\mathcal{K}} x_C \operatorname{cost}(C) \quad \text{s. t.} \tag{M}$$

$$z(E(S)) + \sum_{C\in\mathcal{K}} x_C(|R(C)\cap S| - 1)^+ \le y(S) - y_{max}(S) \qquad \forall S \subseteq V \tag{1}$$

$$z(E) + \sum_{C\in\mathcal{K}} x_C(|R(C)| - 1)^+ = y(V) - 1 \tag{2}$$

$$y_v = 1 \qquad \forall v \in R \tag{3}$$

$$x, y, z \ge \mathbb{0}.$$

Note that if for a feasible solution (x, y, z) to this LP, $z = \mathbb{0}$ and $y_v = 0$ for all $v \in V \setminus R$, then x is a solution to HYP. On the other hand, if $x = \mathbb{0}$, then (y, z) is a solution to BCR. Hence we want to begin with a feasible solution to (M) with $x = \mathbb{0}$, and end with one where $z = \mathbb{0}$ and $y = \chi(R)$, where $\chi(R)$ is the characteristic vector of the terminal set.

Let H be the *support graph* of (y, z), where $V(H) = \{v \in V : y_v > 0\}$ and $E(H) = \{e \in E : z_e > 0\}$. Observe that, whenever there is an edge $uv \in E(H)$ connecting two terminals $u, v \in R$, we may transfer the value z_{uv} to the x variable of the corresponding full component without affecting the feasibility or cost of our solution. We will therefore now assume that H has no edge connecting terminals.

We first describe an algorithm for picking a specific tree C^* in H. Define the *slack* of a vertex set $S \subseteq V$ as

$$\operatorname{sl}(S) := y(S) - y_{max}(S) - z(E(S)) - \sum_{C\in\mathcal{K}} x_C(|R(C)\cap S| - 1)^+,$$

and note that $\operatorname{sl}(S) \ge 0$ in a feasible solution (x, y, z) to (M). We will call a set S *tight* if $\operatorname{sl}(S) = 0$. Furthermore, we denote by $C^*[S]$ the subgraph of C^* induced by the vertices in $S \cap V(C^*)$. We will use Algorithm 1 to compute C^*.

It turns out to be the case that for Steiner claw-free instances, C^* always contains a terminal. We will not assume this for the following analysis however, and so it is convenient to include an "empty" full-component in $\mathcal{K}$, which contains no terminals. Such a component of course has no impact on (M). We also allow full-components containing only a single terminal, which also have no impact on (M). In any case, $C \in \mathcal{K}$ is always a maximal full-component contained in the tree C^*, and $R(C) = R(C^*)$.

Given C^* and full-component C as computed by Algorithm 1, and some $\varepsilon > 0$, we obtain $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ from (x, y, z) by adding ε to x_C, and subtracting ε from y_v and z_e for $v \in V(C^*) \setminus R$ and $e \in E(C^*)$, respectively. Note that this does not increase the cost of the solution. We will argue that if our input instance has no Steiner claw, $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ is feasible for (M) for some choice of $\varepsilon > 0$ small enough. This leads to Algorithm 2.

Note that at the end of the algorithm, $y = \chi(R)$ but also $z = \mathbb{0}$. This is because for every edge uv with $v \notin R$, Constraint (1) on the set $S = \{u, v\}$ implies $z_{uv} \leq y_v = 0$. Moreover, we explicitly moved all z-value from edges between terminals. Hence if the algorithm succeeds, we computed a solution to HYP of the same cost as the solution to BCR we started from.

Algorithm 2 can be implemented in polynomial time. The details of this can be found in Section 4. It is necessary to show that (i) C^* can be efficiently computed for a given solution to (M), (ii) the correct choice of ε in Step 5 of Algorithm 2 can be efficiently found, and (iii) the number of iterations in Algorithm 2 is polynomially bounded. Roughly speaking, (i) follows by reducing the problem of finding a set of minimum slack (under certain restrictions) to a flow problem, and (ii) then follows by applying parametric search methods to this reduction. For point (iii), we obtain a bound of $O(|V|^2)$ on the number of iterations. This is done by arguing via uncrossing techniques that the number of "independent" tight constraints of a certain form for a solution to (M) cannot be too large. A new constraint becomes tight at the start of each iteration, with all previous ones remaining tight, and the bound follows.

In order for Algorithm 2 to produce a feasible HYP solution of no larger cost than the initial BCR solution, we need to show that it is always possible to select $\varepsilon > 0$ at Step 5, while maintaining feasibility for (x, y, z). If ε is small enough, all variables in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ have non-negative values, because C^* is a subgraph of the support graph H of (y, z). Furthermore, going from (x, y, z) to $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ does not change the value of y_v for any terminal $v \in R$, and thus (3) is unaffected by this change. It remains to check that $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ satisfies (1), and moreover that the constraint remains tight for $S = V$, so that (2) is also satisfied. We begin by characterizing when a tight set in (x, y, z) remains feasible in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$.

▶ **Lemma 3.** *Let $S \subseteq V$ be tight in a feasible solution (x, y, z) to* (M)*, C^* be a tree of the support graph H of (y, z), $V(C^*) \cap S \neq \emptyset$, and $\varepsilon > 0$ small enough. Then S is feasible in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ if and only if (i) $C^*[S]$ is connected, and (ii) $\{v \in S : y_v = y_{max}(S)\} \subseteq V(C^*)$ if $S \cap R = \emptyset$, or $R(C^*) \cap S \neq \emptyset$ if $S \cap R \neq \emptyset$. Moreover, S remains tight in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$.*

Proof. First consider the case when $S \cap R = \emptyset$. Let $S_m = \{v \in S : y_v = y_{max}(S)\}$, and define $\rho = 1$ if $S_m \subseteq V(C^*)$, and $\rho = 0$ otherwise. We use $\mathrm{sl}_\varepsilon(S)$ for the slack of set S in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$, and obtain

$$\begin{aligned}\mathrm{sl}_\varepsilon(S) &= \mathrm{sl}(S) + \varepsilon\big(-|V(C^*[S])| + \rho + |E(C^*[S])| - (|R(C^*) \cap S| - 1)^+\big) \\ &= \mathrm{sl}(S) + \varepsilon\big(-|V(C^*[S])| + \rho + |E(C^*[S])|\big).\end{aligned}$$

But since $C^*[S]$ is a forest, $|E(C^*[S])| \leq |V(C^*[S])| - 1$, with equality only if $C^*[S]$ is connected. The result follows.

Algorithm 2 Converting a BCR solution to an HYP solution.

1: Start with a solution (x, y, z) feasible for (M) with $x = \mathbb{0}$.
2: For any $z_{vw} > 0$ with $v, w \in R$, move all weight to the corresponding x variable.
3: **while** $y \neq \chi(R)$ **do**
4: Apply Algorithm 1 to compute a tree C^* and a full-component C.
5: Choose $\varepsilon > 0$ maximally such that $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ remains feasible for (M), and replace (x, y, z) with this new solution.
6: **end while**
7: Output (x, y, z).

Note that if there was a Steiner vertex v with $y_v > 1$, then Constraint (1) for the set V would contradict Constraint (2). In particular this means that in a feasible solution to (M), the terminals have maximal y-values. Hence in the case where $S \cap R \neq \emptyset$ we obtain

$$\mathrm{sl}_\varepsilon(S) = \mathrm{sl}(S) + \varepsilon\big(-|V(C^*[S \setminus R])| + |E(C^*[S])| - (|R(C^*) \cap S| - 1)^+\big).$$

Thus S stays feasible if and only if $|V(C^*[S \setminus R])| + (|R(C^*) \cap S| - 1)^+ \leq |E(C^*[S])|$. Let ρ' be 1 if $R(C^*) \cap S \neq \emptyset$ and 0 otherwise. Then, simplifying further, S stays feasible if and only if $|V(C^*[S])| - \rho' \leq |E(C^*[S])|$. Again since $C^*[S]$ is a forest, $|V(C^*[S])| \geq |E(C^*[S])| + 1$, with equality if and only if $C^*[S]$ is connected. So the inequality is satisfied if and only if $C^*[S]$ is connected and $R(C^*) \cap S \neq \emptyset$, in which case it is satisfied with equality. ◀

Note that the constraint corresponding to V is tight in (x, y, z). Thus if it is feasible in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$, by Theorem 3 it will remain tight, and (2) will be satisfied. The goal is now to apply Theorem 3 to show that $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ satisfies (1) for some $\varepsilon > 0$ whenever there is no Steiner claw. By construction (see Algorithm 1), $C^*[S]$ is connected for all tight sets $S \subseteq V$. Thus we can shift $\varepsilon > 0$ of the value of the y and z variables associated with C^* to x_C, unless there is a tight set *demanding* the inclusion of (some of) its vertices with maximum y-value (e.g. terminals) in C^*.

▶ **Definition 4.** A tight set S for which $V(C^*) \cap S \neq \emptyset$ is called a *demanding set* if $R(C^*) \cap S = \emptyset$ in case S contains a terminal, or if there is some vertex $v \in S \setminus V(C^*)$ for which $y_v = y_{max}(S)$ in case S has no terminals.

3 Analysis of the Algorithm

In this section, we show that if a demanding set S exists, then we can identify a Steiner claw in H. This implies that for Steiner claw-free instances, Algorithm 2 will always find an $\varepsilon > 0$ at Step 5, and therefore terminates successfully. As for the algorithm for quasi-bipartite instances, we will rely on uncrossing arguments of tight sets. However, it will not be sufficient to only uncross tight sets intersecting in terminals. Therefore we develop more advanced uncrossing techniques in our arguments.

Demanding Sets and Blocked Edges

Let S be a demanding set for the tree C^* chosen by Algorithm 1. Thus $S \cap V(C^*) \neq \emptyset$ and $C^*[S]$ is connected.

▶ **Lemma 5.** *Let U be a tight set of a feasible solution (x, y, z) to (M), and let H be the support graph of (y, z). If $U \cap R \neq \emptyset$, then every connected component of $H[U]$ contains a terminal. If $U \cap R = \emptyset$, then $H[U]$ is connected.*

Proof. Assume the statement is wrong. Regardless of whether U contains terminals or not, there must then be a connected component in $H[U]$ with vertex set U_1, such that $U_1 \cap R = \emptyset$ and $U_2 := V(H[U]) \setminus U_1$ is non-empty. In particular, $E(U) = E(U_1) \cup E(U_2)$, $|R(C) \cap U_1| = 0$

for every full component $C \in \mathcal{K}$, and $y_{max}(U_2) > 0$. Thus,

$$\begin{aligned}
&\mathrm{sl}(U_1 \cup U_2) = y(U) - y_{max}(U) - z(E(U)) - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U| - 1)^+ \\
&= y(U_1) + y(U_2) - y_{max}(U_1 \cup U_2) - z(E(U_1)) - z(E(U_2)) \\
&- \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_2| - 1)^+ \\
&> y(U_1) - y_{max}(U_1) - z(E(U_1)) - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_1| - 1)^+ \\
&\quad + y(U_2) - y_{max}(U_2) - z(E(U_2)) - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_2| - 1)^+ \\
&= \mathrm{sl}(U_1) + \mathrm{sl}(U_2).
\end{aligned}$$

By feasibility of U_1 and U_2, $U_1 \cup U_2$ cannot be tight, a contradiction. ◀

Consider a path P_1 in $H[S]$ for the demanding set S, that connects $S \cap V(C^*)$ to some vertex $v \in S \setminus V(C^*)$ with $y_v = y_{max}(S)$ (e.g. a terminal). By Theorem 5 this path exists, whether or not S contains terminals. Traversing the path from v, let b_1 be the first vertex of C^*, and let a_1 be its immediate predecessor (see Figure 3). Note that b_1 must be a Steiner vertex, otherwise S would not be a demanding set. We will in fact be able to show later that a_1 must also be a Steiner vertex.

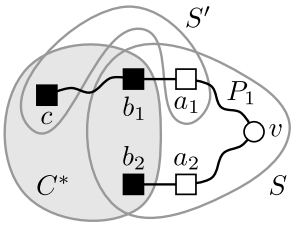


Figure 3 Interaction of a demanding set S and a blocking set S'.

Edge a_1b_1 is called a *blocked* edge: its endpoint b_1 is part of C^*, but a_1 was not added to C^* by Algorithm 1. Thus, there must be a tight set S' for which $C^* \cup \{a_1b_1\}$ is disconnected in S', and thus *blocks* the addition of a_1b_1; we call S' a *blocking set*. S' contains a_1, not b_1, but some other vertex $c \in V(C^*)$. The following technical lemma helps us to argue that a demanding set must have two distinct blocked edges. In its statement, we use $\delta_H(A, B)$ for the collection of edges of H with one endpoint in vertex set A, and the other in B.

▶ **Lemma 6.** *Let U be a tight set with a partition $\{U_1, U_2\}$ such that $|\delta_H(U_1, U_2)| = 1$, $y_{max}(U_1) \geq y_{max}(U_2)$, and $U_2 \cap R = \emptyset$. If $u_1u_2 \in \delta_H(U_1, U_2)$ where $u_2 \in U_2$, then $z_{u_1u_2} = y_{u_2}$. Moreover, U_1 and U_2 are tight sets.*

Proof. Let $\mathcal{K}_{12} = \{C \in \mathcal{K} : R(C) \cap U_1 \neq \emptyset \wedge R(C) \cap U_2 \neq \emptyset\}$ be those full-components that intersect both U_1 and U_2. Noting that $y_{max}(U) = y_{max}(U_1)$, consider the slack of U_1:

$$\begin{aligned}
&\mathrm{sl}(U_1) = \mathrm{sl}(U_1) + \mathrm{sl}(U_2) - \mathrm{sl}(U_2) \\
&= y(U_1) + y(U_2) - y_{max}(U_1) - y_{max}(U_2) - z(E(U_1)) - z(E(U_2)) \\
&\quad - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_1| - 1)^+ - \sum_{C\in\mathcal{K}} x_C(|R(C) \cap U_2| - 1)^+ - \mathrm{sl}(U_2) \\
&\quad = \mathrm{sl}(U) - y_{max}(U_2) + z_{u_1u_2} + \sum_{C\in\mathcal{K}_{12}} x_C - \mathrm{sl}(U_2).
\end{aligned}$$

We know that U is tight so that $\mathrm{sl}(U) = 0$, and our solution is feasible which means $\mathrm{sl}(U_2) \geq 0$. Also there are no terminals in U_2 so that $\sum_{C\in\mathcal{K}_{12}} x_C = 0$. From Constraint (1) on the set $\{u_1, u_2\}$ we get $z_{u_1u_2} \leq y_{u_2} \leq y_{max}(U_2)$. Hence

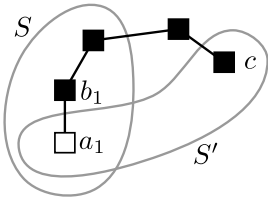
$$\mathrm{sl}(U_1) = z_{u_1u_2} - y_{max}(U_2) - \mathrm{sl}(U_2) \leq 0.$$

By feasibility, $\mathrm{sl}(U_1) \geq 0$ and therefore $\mathrm{sl}(U_1) = 0$. Moreover, the above inequality can only be satisfied with equality if $\mathrm{sl}(U_2) = 0$ and $z_{u_1u_2} = y_{max}(U_2)$. This means that also U_2 is tight, and $z_{u_1u_2} = y_{u_2} = y_{max}(U_2)$, which proves the claim. ◀

The above lemma enables us to prove that there must be a second blocked edge $a_2b_2 \in H[S]$, $a_1b_1 \neq a_2b_2$, that crosses from $S \setminus V(C^*)$ to $S \cap V(C^*)$.

▶ **Lemma 7.** *Every demanding set S has at least two blocked edges.*

Proof. Suppose that there is a single edge $a_1b_1 \in \delta_H(S \setminus V(C^*), S \cap V(C^*))$, for the sake of contradiction. Since S is a demanding set, we have $y_{max}(S \setminus V(C^*)) \geq y_{max}(S \cap V(C^*))$ and $S \cap V(C^*) \cap R = \emptyset$. Thus, by Theorem 6, $z_{a_1b_1} = y_{b_1}$.



Now consider the set $S' \cup \{b_1\}$ where S' is the blocking set for a_1b_1. Since $a_1 \in S'$ and $b_1 \notin S'$, this set includes the edge a_1b_1, while S' does not. We know that S' is tight and therefore Constraint (1) on $S' \cup \{b_1\}$ can only be feasible if a_1b_1 is the *only* edge in H added to this set, i.e. $E(S' \cup \{b_1\}) \setminus E(S')$ contains only a_1b_1. Moreover, for the same reason $S' \cup \{b_1\}$ must be tight. Also, S' contains some other vertices of $V(C^*)$, which cannot be adjacent to b_1 as otherwise $E(S' \cup \{b_1\}) \setminus E(S')$ would contain more than one edge. Hence $S' \cup \{b\}$ is a tight set for which $C^*[S' \cup \{b_1\}]$ is disconnected. This contradicts our construction of C^*. ◀

To show that the vertices a_1, a_2, b_1, b_2, and c that we have found can be used to construct a Steiner claw, we need to show that they are Steiner vertices. For this we will analyze the intersections of demanding sets and their blocking sets. In particular we show next that the intersection does not contain any terminal. Note that for our main result we can assume w.l.o.g. that the considered demanding set is inclusion-wise minimal.

▶ **Lemma 8.** *Let S be an inclusion-wise minimal demanding set, and S' a blocking set for S. Then $S \cap S' \cap R = \emptyset$.*

To prove this lemma we need the following standard fact about tight sets sharing terminals.

▶ **Lemma 9.** *For any feasible solution (x, y, z) to (M), suppose U_1 and U_2 are tight sets, such that $U_1 \cap U_2 \cap R \neq \emptyset$. Then $U_1 \cap U_2$ and $U_1 \cup U_2$ are also tight. Also, (i) $\delta_H(U_1 \setminus U_2, U_2 \setminus U_1) = \emptyset$, where H is the support graph of (y, z), and (ii) for all $C \in \mathcal{K}$ with $x_C > 0$ and $R(C) \cap U_i \neq \emptyset$ for both $i \in \{1, 2\}$, $R(C) \cap U_1 \cap U_2 \neq \emptyset$.*

Proof. Since $U_1 \cap U_2 \cap R \neq \emptyset$, we have that

$$y_{max}(U_1) = y_{max}(U_2) = y_{max}(U_1 \cap U_2) = y_{max}(U_1 \cup U_2) = 1.$$

We also have that $S \to z(E(S))$ and $S \to (|R(C) \cap S| - 1)^+$ are both supermodular functions, which means that

$$z(E(U_1)) + z(E(U_2)) \leq z(E(U_1 \cup U_2)) + z(E(U_1 \cap U_2)), \tag{4}$$

and, for any $C \in \mathcal{K}$,

$$(|R(C) \cap U_1| - 1)^+ + (|R(C) \cap U_2| - 1)^+ \leq (|R(C) \cap (U_1 \cap U_2)| - 1)^+ + (|R(C) \cap (U_1 \cup U_2)| - 1)^+. \tag{5}$$

Hence

$$\mathrm{sl}(U_1 \cup U_2) + \mathrm{sl}(U_1 \cap U_2) \leq \mathrm{sl}(U_1) + \mathrm{sl}(U_2) = 0. \tag{6}$$

Each term on the left-hand side is non-negative by feasibility, and thus $U_1 \cap U_2$ and $U_1 \cup U_2$ are tight as well.

For the second part, since (6) holds with equality, (4) must hold with equality as well. This can only be if $z(\delta(U_1 \setminus U_2, U_2 \setminus U_1)) = 0$. Whenever $x_C > 0$, also (5) must hold with equality. If $R(C) \cap U_i \neq \emptyset$ for both $i \in \{1, 2\}$ then this is only possible if $R(C) \cap U_1 \cap U_2 \neq \emptyset$. ◀

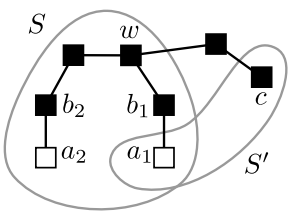
Proof of Theorem 8. Assume the claim is wrong and the intersection of S and S' contains a terminal. Consider the case when S and S' do not intersect in $V(C^*)$. Note however that both sets contain vertices of $V(C^*)$. Let $U_1 = S \cap V(C^*)$ and $U_2 = S' \cap V(C^*)$. Since $\delta_H(S \setminus S', S' \setminus S) = \emptyset$ by Theorem 9, no vertex in U_1 is adjacent to a vertex in U_2. But by the same lemma $S \cup S'$ is a tight set in which C^* is disconnected. This contradicts our construction of C^*.

Hence it must be that $S \cap S' \cap V(C^*) \neq \emptyset$. In this case we consider the set $S \cap S'$, which we know is tight by Theorem 9. We also know that one of the vertices incident to the edge that S' blocks in S is not in S', i.e. there is a vertex $b \in S$ such that $b \notin S \cap S'$. Hence $S \cap S'$ is a strict subset of S, which contains no terminal of C^*. However it does contain some terminal and a vertex from $V(C^*)$, and is therefore a demanding set. This contradicts the minimality of S. ◀

Using the insight of Theorem 8 we can finally prove that a demanding set implies the existence of a Steiner claw. We distinguish the cases of whether the demanding set and its blocking set intersect in C^*.

▶ **Lemma 10.** *Let S be an inclusion-wise minimal demanding set, and S' a blocking set for S. If $S \cap S' \cap V(C^*) = \emptyset$ then there is a Steiner claw.*

Proof. By Theorem 7, S has two blocked edges a_1b_1 and a_2b_2 with $a_i \notin V(C^*)$ and $b_i \in V(C^*)$ for $i \in \{1, 2\}$. We know that for $i \in \{1, 2\}$, b_i must be a Steiner vertex since $S \cap R(C^*) = \emptyset$, and the same is true for a_i by Theorem 8. Let S' be a blocking set for a_1b_1. Recall that in Algorithm 1 we first add Steiner vertices and only then terminals. Since S' blocks the addition of the Steiner vertex a_1, it was already a blocking set in Step 2 of the algorithm, before any terminals had been added to C^*. Thus there exists some Steiner vertex $c \in V(C^*) \cap S'$. Also note that by the assumptions of the lemma, $c \notin S$.

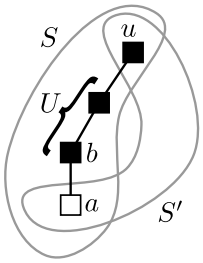


Since $C^*[S]$ is connected, there must be a path P on the Steiner vertices of $C^*[S]$ from b_1 to b_2. Every Steiner vertex of P has at least two Steiner neighbors, since a_1 and a_2 also are Steiner vertices and are neighbors to the endpoints of P. Note that c cannot be part of P since c is not in S. However it is in the component C^*. Let w be the first vertex of P reached by the unique path Q in C^* from c to P. The path Q has non-zero length, and since a_1 and a_2 are not in C^*, Q contains neither a_1 nor a_2. Moreover, Q contains only Steiner vertices since c also is a Steiner vertex. Hence w has at least three Steiner neighbors: two since it is in P and an additional one from Q. ◀

We are left with the case where demanding and blocking sets intersect in C^*. Hence the following lemma completes the proof of correctness for Algorithm 2, and therefore also that of Theorem 1.

▶ **Lemma 11.** *Let S be an inclusion-wise minimal demanding set, and S' a blocking set for S. If $S \cap S' \cap V(C^*) \neq \emptyset$ then there is a Steiner claw.*

Proof. Assume the claim is false, so that the intersection of S and S' contains a vertex of C^* and every Steiner vertex has at most two Steiner neighbors. Let ab be an edge blocked by S'. Consider a path P in $H[S]$ starting with ab and continuing from b along vertices of C^* until a vertex $u \in S \cap S'$ is reached, i.e. $V(P) \cap S' = \{a, u\}$ and $V(P) \setminus V(C^*) = \{a\}$. Such a path exists since $C^*[S]$ is connected.



Let $U = V(P) \setminus S'$ be the vertices on P exclusively in S. Note that this set is non-empty since $b \in U$, and it contains only Steiner vertices since $S \cap R(C^*) = \emptyset$. Our goal is to show that the z-values of $E(P)$ are small compared to the y-values of U. As a consequence we will see that removing U from S gives a demanding set, thus contradicting the minimality of S.

The edge set $E(S' \cup U)$ is a superset of $E(S') \cup E(P)$. Thus from Constraint (1) on the set $S' \cup U$ we can conclude that

$$\begin{aligned} &z(E(S')) + z(E(P)) + \sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S' \cup U)| - 1)^+ \\ &\leq z(E(S' \cup U)) + \sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S' \cup U)| - 1)^+ \\ &\qquad \leq y(S' \cup U) - y_{max}(S' \cup U) = y(S') + y(U) - y_{max}(S' \cup U). \end{aligned}$$

Note that $\sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S' \cup U)| - 1)^+ = \sum_{C \in \mathcal{K}} x_C(|R(C) \cap S'| - 1)^+$ since U contains only Steiner vertices. Hence substituting $z(E(S'))$ from Constraint (1) on the tight set S' in the above inequality and eliminating superfluous terms gives $z(E(P)) \leq y(U) + y_{max}(S') - y_{max}(S' \cup U) \leq y(U)$.

We now remove U from S and bound $z(E(S \setminus U))$. Consider an edge $vw \in E(S)$ where $v \in U$ and $w \in S \setminus (U \cup \{u, a\})$. Since any Steiner vertex from U has at most two Steiner neighbors, w must be a terminal, i.e. a maximum valued vertex in S. However S does not contain any terminals of C^* and therefore vw must be a blocked edge for S. This also means that there is a blocking set S'' preventing w to be part of C^*. However by Theorem 8, S and S'' do not share terminals. Hence such an edge vw cannot exist. This means that the edges in $E(S)$ can be partitioned into $E(S \setminus U)$ and $E(P)$ and therefore $z(E(S \setminus U)) = z(E(S)) - z(E(P))$.

Some vertex v with maximum value in S lies outside of C^* and is therefore not contained in U, and thus $y_{max}(S) = y_{max}(S \setminus U)$. Note that $\sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S \setminus U)| - 1)^+ = \sum_{C \in \mathcal{K}} x_C(|R(C) \cap S| - 1)^+$, as above. Hence the tightness of S together with the inequality $z(E(P)) \leq y(U)$ gives

$$\begin{aligned} z(E(S)) - z(E(P)) &\geq y(S) - \sum_{C \in \mathcal{K}} x_C(|R(C) \cap S| - 1)^+ - y_{max}(S) - y(U) \\ &= y(S \setminus U) - y_{max}(S \setminus U) - \sum_{C \in \mathcal{K}} x_C(|R(C) \cap (S \setminus U)| - 1)^+. \end{aligned}$$

Due to Constraint (1) on $S \setminus U$ of our feasible solution, this implies that $S \setminus U$ must be tight. However this set is a strict subset of S, intersects $V(C^*)$ (for instance at u), and it contains the maximum valued vertex v, which is not in C^*. Hence S was not an inclusion-wise minimal demanding set, which is a contradiction. ◀

4 Algorithmic Issues

In order to show that Algorithm 2 can be implemented efficiently, we need to show that (i) the number of iterations of the algorithm is polynomial, and (ii) that we can compute the correct choice of C^* in each iteration, and the amount that we should extract.

4.1 Bounding the Number of Iterations

We prove the following:

▶ **Theorem 12.** *Given a Steiner tree instance with n nodes and m edges, the number of iterations of Algorithm 2 is $O(n^2)$.*

Let the Steiner tree instance be described by $G = (V, E)$ and terminal set R. Let (y^0, z^0) be the initial solution to BCR, which we extend to a solution (x^0, y^0, z^0) of HYP with $x^0 = \mathbb{0}$. Let (x^i, y^i, z^i) denote the solution obtained after i iterations, i.e., i components have been maximally extracted. Let i_{max} denote the index of the final iteration, so $y^{i_{max}} = \chi(R)$ and $z^{i_{max}} = \mathbb{0}$.

We will first observe that once a set becomes tight, it remains tight from then on.

▶ **Lemma 13.** *For all $i \leq j$, if $S \subseteq V$ is tight in iteration i, then it is tight in iteration j.*

Proof. An immediate corollary of Theorem 3. ◀

It is also clear that if $i \leq j$, then $z^i_e = 0$ implies that $z^j_e = 0$, and $y^i_v = 0$ implies that $y^j_v = 0$. At the end of each iteration, a new constraint must become tight, and this constraint must be independent of, i.e. not implied by, the previously tight constraints. So in order to bound the number of iteration, it is enough to show that the number of independent tight constraints can never be too large. This we will show via standard combinatorial uncrossing arguments, albeit with some technicalities.

Let $\mathcal{K}' = \{C \in \mathcal{K} : x^{i_{max}}_C > 0\}$. Let $\mathcal{R} = 2^V \dot\cup E$ denote the set of constraints of (M) corresponding to (1) and the nonnegativity constraints for z. For any $\ell \in \mathcal{R}$, let $\Gamma(\ell)$ denote row ℓ of the constraints matrix of (M); so $\Gamma(\ell)$ is a vector in $\mathbb{R}^{V \cup E \cup \mathcal{K}'}$.

▶ **Lemma 14.** *For any $i \in [i_{max}]$, and any two sets S_1, S_2 with $S_1 \cap S_2 \cap R \neq \emptyset$ that are tight in iteration i,*

$$\Gamma(S_1) + \Gamma(S_2) - \Gamma(S_1 \cup S_2) - \Gamma(S_1 \cap S_2) \in \operatorname{span}(\{\Gamma(e) : z^i_e = 0\}).$$

Proof. First, since S_1 and S_2 remain tight in the final iteration, and $x^{i_{max}}_C > 0$ for all $C \in \mathcal{K}'$, we may deduce from Theorem 9 applied to $(x^{i_{max}}, y^{i_{max}}, z^{i_{max}})$ that there are no components "crossing" S_1 and S_2, meaning that if $R(C)$ intersects both S_1 and S_2, it must intersect $S_1 \cap S_2$. It follows that for any $C \in \mathcal{K}'$,

$$f_C(S_1) + f_C(S_2) = f_C(S_1 \cup S_2) + f_C(S_1 \cap S_2),$$

where $f_C(S) := (|R(C) \cap S| - 1)^+ = |R(C) \cap S| - [R(C) \cap S \neq \emptyset]$ is the coefficient of x_C for the constraint corresponding to S in (M).

Let $F = \{e \in \delta(S_1 \setminus S_2, S_2 \setminus S_1) : z^i_e = 0\}$. We may deduce from Theorem 9, this time applied to (x^i, y^i, z^i), that $z^i(\delta(S_1 \setminus S_2, S_2 \setminus S_1)) = 0$. Hence

$$\chi(E(S_1)) + \chi(E(S_2)) = \chi(E(S_1 \cup S_2)) + \chi(E(S_1 \cap S_2)) + \chi(F).$$

Since also $\chi(S_1) + \chi(S_2) = \chi(S_1 \cup S_2) + \chi(S_1 \cap S_2)$, the lemma follows. ◀

▶ **Lemma 15.** *Fix any $i \in [i_{max}]$. Let $\mathcal{R}_{tight} \subseteq \mathcal{R}$ be the subset of $\mathcal{R}$ that are tight constraints in (x^i, y^i, z^i). Then*

$$\dim \operatorname{span}(\{\Gamma(\ell) : \ell \in \mathcal{R}_{tight}\}) = O(n^2).$$

Proof. Let $\mathfrak{F} = \{\Gamma(e) : e \in E, z^i_e = 0\}$. Fix any $r \in R$, and let

$$\mathfrak{R}_r = \{\Gamma(S) : r \in S \text{ and } \operatorname{sl}(S) = 0\} \cup \mathfrak{F}.$$

Let $\mathcal{L}_r$ be a maximal laminar family of tight sets in (x^i, y^i, z^i) containing r; so in fact, $\mathcal{L}_r$ is a chain. We claim that

$$\operatorname{span}(\{\Gamma(S) : S \in \mathcal{L}_r\} \cup \mathfrak{F}) = \operatorname{span}(\mathfrak{R}_r).$$

This follows immediately from Theorem 14 by an argument of Jain [13]. If for some tight set U with $r \in U$, $\Gamma(U)$ was not in $\operatorname{span}(\{\Gamma(S) : S \in \mathcal{L}_r\} \cup \mathfrak{F})$, we could uncross S w.r.t. $\mathcal{L}_r$ to obtain a strictly larger laminar family, a contradiction.

Applying this reasoning for each $r \in R$, we conclude that

$$\operatorname{span}\Big(\Big\{\Gamma(S) : S \in \bigcup_{r \in R} \mathcal{L}_r\Big\} \cup \mathfrak{F}\Big) = \operatorname{span}(\Gamma(\ell) : \ell \in \mathcal{R}_{\text{tight}}).$$

Since $|\mathcal{L}_r| = O(n)$ and $|\mathfrak{F}| = O(m)$, the result follows. ◀

Theorem 12 is proven.

4.2 Determining the Minimal Tight Sets, and the Duration of Each Iteration

The main observation here will be that checking if a solution (x, y, z) is feasible for (M), as well as checking for tight sets under certain constraints, can be reduced to solving certain maximum flow problems. This will allow for the efficient determination of the component C^* for each iteration, as well as the duration of each iteration using parametric search methods. The construction extends one for HYP described in [11] (as well as classical results for separation over the forest polytope); no major new ideas are needed, though for convenience some aspects of the construction are different.

We construct the directed graph $D = (W, A)$ with capacities ξ as follows. Let $W = V \cup \{r_C : C \in \mathcal{K}, x_C > 0\} \cup \{s, t\}$, where r_C is a new vertex for each component C, and s and t will be source and sink vertices. Let $M = \sum_{C \in \mathcal{K}} x_C$. For each $e \in \operatorname{supp}(z)$, add both orientations of the edge to A, giving both arcs capacity $\frac{1}{2}z(e)$; for each $r_C \in W \setminus V$, add an arc of capacity x_C from r_C to t, and infinite capacity arcs from each terminal in $R(C)$ to r_C. For each $v \in V$, add the arc sv with capacity $M + \frac{1}{2}z(\delta(v))$, and the arc vt with capacity $M + y_v - \sum_{C \in \mathcal{K} : v \in R(C)} x_C$. The role of M is solely to ensure that all capacities are nonnegative.

▶ **Theorem 16.** *Let S, T be two disjoint subsets of V, with S nonempty and satisfying $\max_{w \in S} y_w = \max_{w \in V \setminus T} y_w$. Given a (feasible or infeasible) solution (x, y, z) to (M), a set $U^* \subseteq V$ is of minimal slack under the constraint $S \subseteq U^* \subseteq (V \setminus T)$ if and only if $U^* \cup \{r_C \in W \setminus V : R(C) \cap S \neq \emptyset\}$ is a minimum capacity $(\{s\} \cup S)$-$(\{t\} \cup T)$-cut in D.*

Note that, for example, in order to find an overall minimal slack set U^*, one may first guess $w \in V$ s. t. $y_w = y_{max}(U^*)$. Then apply the above theorem with $T = \{v \in V : y_v > y_w\}$ and $S = \{w\}$. Trying all possibilities for w, U^* can be found with n maximum flow computations.

Proof. Observe that if Q is an $(\{s\} \cup S)$-$(\{t\} \cup T)$-cut in D, but with $r_C \notin Q$ for some $C \in \mathcal{K}$ where $R(C) \cap Q \neq \emptyset$, then $\xi(\delta_D^+(Q)) = \infty$. Conversely, if $r_C \in Q$ but $R(C) \cap Q = \emptyset$, then removing r_C from Q yields a cut of strictly smaller capacity.

So consider any $(\{s\} \cup S)$-$(\{t\} \cup T)$-cut Q satisfying $\{C \in \mathcal{K} : r_C \in Q\} = \{C \in \mathcal{K} : R(C) \cap Q \neq \emptyset\}$. Let $U = Q \cap V$. Then

$$\begin{aligned}\xi(\delta_D^+(Q)) &= \sum_{v \in U} \Big(M + y_v - \sum_{C \in \mathcal{K} : v \in R(C)} x_C \Big) + \tfrac{1}{2} z(\delta_G(U)) \\ &\quad + \sum_{v \in V \setminus U} \Big(M + \tfrac{1}{2} z(\delta_G(\{v\})) \Big) + \sum_{C \in \mathcal{K} : C \cap R(U) \neq \emptyset} x_C \\ &= M \cdot |V| + y(U) + z(E) - z(E(U)) - \sum_{C \in \mathcal{K}} x_C (|R(C) \cap U| - 1)^+ \\ &= \mathrm{sl}(U) + M \cdot |V| + y_{max}(U) + z(E).\end{aligned}$$

By the conditions on S and T, $y_{max}(U) = \max_{w \in S} y_w$. Thus all terms in the above aside from $\mathrm{sl}(U)$ are independent of U. The result follows. ◀

Choosing C^*

Given a solution (x, y, z) to (M) and any Steiner vertex ℓ with $y_\ell > 0$, we will now show how the choice of C^* described in Section 2 can be efficiently computed.

Suppose we are considering adding $v \in V$ to our current C^*, with $vu \in \mathrm{supp}(z)$ and $u \in V(C^*) \setminus R$. (Here, v could be either a Steiner node, if we are in step 2, or a terminal if we are in step 3.) Let C' be the component obtained by adding v and vu to C^*. The only reason to not add v is that there is some tight set U for which C' would be disconnected in U. By assumption, C^* is connected in U. Thus $u \notin U$, and $v \in U$. By trying all possibilities for w which might be a maximizer of y in U, and hence applying Theorem 16 with $S = \{w, v\}$ and $T = \{u\} \cup \{v' \in V : y_{v'} > y_w\}$, we can determine whether such a tight set U exists or not, and hence whether v should be added to C^*.

The Choice of ε in an Iteration

What remains is to determine what value ε should take in a particular iteration. Let (x, y, z) denote the solution at the start of the iteration, and let $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ denote the solution after an amount ε of the current component C^* has been extracted. As before, let $\mathrm{sl}_\varepsilon(S)$ denote the slack of set S in $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$.

It is of course easy to determine the maximum value of ε such that all nonnegativity constraints remain satisfied. So the main challenge is to determine ε such that a new tight set U forms (which would then be violated if a larger value of ε was chosen). It is clearly sufficient to compute, for each $w \in V$, the maximum value of ε such that $\min_{U \subseteq V : y_w = y_{max}(U)} \mathrm{sl}_\varepsilon(U) \geq 0$. (We may then simply take the minimum over all the values of ε obtained).

The maximum flow problem we have constructed has capacities that are linear functions of (x, y, z). Moreover, $(x(\varepsilon), y(\varepsilon), z(\varepsilon))$ is a linear function of ε. Thus a parametric maximum flow algorithm can be applied [16].

References

1 A. Borchers and D. Du. The k-Steiner ratio in graphs. *SIAM Journal on Computing*, 26(3):857–869, 1997.

2 J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM*, 60(1):6:1–6:33, 2013.

3 D. Chakrabarty, J. Könemann, and D. Pritchard. Hypergraphic LP relaxations for Steiner trees. In *International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 383–396, 2010.

4 D. Chakrabarty, J. Könemann, and D. Pritchard. Integrality gap of the hypergraphic relaxation of steiner trees: A short proof of a 1.55 upper bound. *Operations Research Letters*, pages 567–570, 2010.

5 M. Chlebík and J. Chlebíková. Approximation hardness of the Steiner tree problem on graphs. In *Proceedings, Scandinavian Workshop on Algorithm Theory*, pages 170–179, 2002.

6 J. Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards B*, 71B:233–240, 1967.

7 J. Edmonds. Matroids and the greedy algorithm. *Math. Programming*, 1:127–136, 1971.

8 DIMACS Center for Discrete Mathematics and Theoretical Computer Science. 11th DIMACS implementation challenge in collaboration with ICERM: Steiner tree problems. `http://dimacs11.cs.princeton.edu/`, 2014.

9 I. Fung, K. Georgiou, J. Könemann, and M. Sharpe. Efficient algorithms for solving hypergraphic steiner tree relaxations in quasi-bipartite instances. *CoRR*, abs/1202.5049, 2012.

10 M. X. Goemans and Y. Myung. A catalog of Steiner tree formulations. *Networks*, 23(1):19–28, 1993.

11 M. X. Goemans, N. Olver, T. Rothvoß, and R. Zenklusen. Matroids and integrality gaps for hypergraphic steiner tree relaxations. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1161–11762, 2012.

12 F.K. Hwang, D.S. Richards, and P. Winter. *The Steiner tree problem*. Monograph in Annals of Discrete Mathematics, 53. Elsevier, 1992.

13 K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 448–457, 1998.

14 R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, NY, 1972.

15 J. Könemann, D. Pritchard, and K. Tan. A partition-based relaxation for Steiner trees. *Math. Programming*, 127(2):345–370, 2011.

16 N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *Journal of the ACM*, 30(4):852–865, 1983.

17 T. Polzin and S. Vahdati-Daneshmand. On Steiner trees and minimum spanning trees in hypergraphs. *Operations Research Letters*, 31(1):12–20, 2003.

18 S. Rajagopalan and V. V. Vazirani. On the bidirected cut relaxation for the metric Steiner tree problem. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 742–751, 1999.

19 D. Warme. *Spanning Trees in Hypergraphs with Applications to Steiner Trees*. PhD thesis, University of Virginia, 1998.

20 R. T. Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Math. Programming*, 28:271–287, 1984.

Reaching Consensus via Non-Bayesian Asynchronous Learning in Social Networks

Michal Feldman[*1], **Nicole Immorlica**[2], **Brendan Lucier**[2], **and S. Matthew Weinberg**[†3]

1 **Tel-Aviv University**
michal.feldman@cs.tau.ac.il
2 **Microsoft Research**
{nicimm,brlucier}@microsoft.com
3 **MIT**
smweinberg@csail.mit.edu

Abstract

We study the outcomes of information aggregation in online social networks. Our main result is that networks with certain realistic structural properties avoid information cascades and enable a population to effectively aggregate information. In our model, each individual in a network holds a private, independent opinion about a product or idea, biased toward a ground truth. Individuals declare their opinions asynchronously, can observe the stated opinions of their neighbors, and are free to update their declarations over time. Supposing that individuals conform with the majority report of their neighbors, we ask whether the population will eventually arrive at consensus on the ground truth. We show that the answer depends on the network structure: there exist networks for which consensus is unlikely, or for which declarations converge on the incorrect opinion with positive probability. On the other hand, we prove that for networks that are sparse and expansive, the population will converge to the correct opinion with high probability.

1998 ACM Subject Classification G.2.2 [Discrete Mathematics]: Graph Theory—Graph Algorithms, G.3 [Probability and Statistics]: Stochastic Processes

Keywords and phrases Information Cascades, Social Networks, non-Bayesian Asynchronous Learning, Expander Graphs, Stochastic Processes

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.192

1 Introduction

A community consists of a collection of individuals, each with their own observations and inferences. Through social interactions, these individuals combine these private reflections with the public opinions of others to form their personal public opinions regarding matters of importance. For many such matters, individuals have aligned goals. Thus, there is often a ground truth, a correct answer, to such questions. When ground truth exists and when individuals' observations are more likely to lead to correct inferences than incorrect ones, the law of large numbers states that a majority of individuals, when reasoning privately, will reach correct conclusions. This leaves a potentially substantial fraction of society with the

* Michal Feldman is partially supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement number 337122.
† Supported by a Microsoft Research Fellowship.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 192–208

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

incorrect conclusion, but it offers hope that the correct majority might influence the society creating a consensus on the ground truth.

Unfortunately, the outcome of this process of social deliberation can result in egregious errors in which the potentially small incorrect minority opinion infiltrates the entire community as individuals copy this opinion. A situation like this, in which individuals copy opinions of others while ignoring their own observations, is called an *information cascade*. Information cascades notoriously block information aggregation. That is, although society has enough information for *everyone* to make the right decision with high probability, there is a substantial chance that everyone makes the wrong decision!

In this work we are motivated by occurrences like the following two real historical events. In the 1930s, the United States experienced a severe drought, spurring a great innovation in agriculture: hybrid corn. These new hybrids offered a yield 15–20 % greater than the open-pollinated varieties, and by the early 40s they dominated the corn belt. Interviews with farmers regarding their adoption practices suggest that the two main factors in the acceptance or rejection of hybrid corn were personal experimentation and the opinions of friends. As farmers repeatedly weighed these factors from year to year, the farming society as a whole gradually began to herd on the highly beneficial decision to plant hybrid corn. In the late 2000s, the United States experienced another period of economic decline that has come to be known as the Great Recession. The cause of the recession is commonly attributed to the collapse of the housing bubble. Economists have argued that, again, a main factor in investors' actions in the context of the housing market was the investment decisions of others. Thus, again the individuals in the community herded on a certain behavior albeit this time a suboptimal one.

How is it that both communities – farmers in the 30s and 40s, and investors in the 2000s – reached agreement on the answer to important questions facing them? How is it that the farmers reached the correct conclusion while the bankers were fooled *en masse*? A crucial difference between these two cases is the structure of the network over which information spreads. Farmers live in local communities and mainly interact with geographically close neighbors whereas investors observe investment decisions of most others.

In the present day, the proliferation of online social platforms such as Facebook and Twitter serves to remove friction in the dissemination of information. One might expect that adoption of new technologies or opinions, in the spirit of hybrid corn or housing investments, would occur at a more rapid pace as a result and have widespread impact. This leads us to our main motivating question: does the structure of large, online social networks enable the efficient aggregation of information, while resisting the proliferation of incorrect beliefs?

An important research question is to understand the factors that influence information cascades. What networks of social interactions, and what patterns of opinion formation allow entire societies to converge on the correct decision? There is a long literature on the topic of social learning, focused on two different barriers for information aggregation: information suppression and information loss. Some models, like standard rational Bayesian learning models [3, 5, 13, 2, 1], capture the information suppression problem. Opinions are private and are only revealed over time. In such a model there might never be, at any time, enough public information in the society to correctly aggregate information. The typical conclusion is that this suppression effect is worst for the complete network, and can be avoided if the network is (in some sense) sparse [2, 1].

In other models, such as repeated synchronous majority dynamics, agents begin by announcing their opinions publicly, so a central observer would initially be able to deduce the correct decision. However, since the agents use heuristics that are based on their own

local view of the network to update their beliefs (e. g., switching to the majority report of one's neighbors), the community might diverge from this state, experiencing information loss. Indeed, there are scenarios in which a very small minority opinion can ultimately dominate the ground truth [4]. However, social learning *does* occur in such models if the network is sufficiently well-connected, and no single individual is too influential [12].

These two lines of work arrive at very different conclusions about the impact of network topology. Our work considers a setting that exhibits both barriers simultaneously. Our model thus captures the tension between two requirements: being sparse enough to prevent information suppression, while being sufficiently connected to prevent information loss.

In our model, the decision at hand is binary, e. g., whether or not to adopt a certain technology. There is a correct decision, and each individual has a (conditionally independent) signal regarding this decision which is more likely to be correct than incorrect (i. e. is correct with probability $1/2 + \delta$ for some $\delta > 0$). Initially, individuals are not stating opinions (as in standard models of rational learning). Individuals are asked to state an opinion, repeatedly and asynchronously.[1] When stating an opinion, individuals simply copy the majority opinion among their friends, breaking ties in favor of their private signal. Our model therefore combines a non-Bayesian update method with the asynchronicity typical of Bayesian models. This asynchronous model is natural in settings of local communication in a population, where the sharing of information is not globally coordinated. We ask: do these asynchronous majority dynamics result in a correct consensus with high probability, for graphs that exhibit realistic properties of large social networks?

We focus on two key features of large social networks. First, they tend to be *expansive*, meaning roughly that they do not contain very sparse cuts. While it has been observed that small social networks tend to have sparse cuts corresponding to divisions between sub-communities, this tends not to be the case for empirically-observed large social networks [10, 11]. Intuitively, expansiveness leads to information diffusion which allows society to reach consensus. Second, social networks tend to be *sparse*. Intuitively, sparsity should limit the rate at which a single individual's opinion can spread in the network, leading to the spread of many independent opinions, and independent opinions are good for producing correct majorities. These two features together thus have some chance of producing a correct consensus so long as the low sparsity allows enough independent decisions to be reached before the high expansiveness takes over diffusing these opinions.

As we show in the full version of the paper, it is not always the case that low average degree is sufficient to build a population to a correct consensus opinion. In fact, it is not even the case that this property suffices to reach a correct majority opinion. In Appendix A of the full version, we provide an example of a network with constant average degree, for which the population will reach a majority on the incorrect opinion with positive probability. The key issue in this construction is the presence of a large clique; that is, while the network is sparse in a global sense, it is not "locally sparse" in the sense that it contains a reasonably large dense subgraph.

Motivated by this example, we turn to stronger notions of sparsity. Specifically, we study the class of expanders with maximum degree d. Our main result is that for any fixed $d > 1$ and a growing family of graphs with maximum degree d and sufficiently high expansiveness, the dynamics described above will reach consensus on the ground truth with high probability.

[1] To our knowledge, this is the first paper to study non-Bayesian asynchronous learning.

▶ **Informal Theorem (Informal) 1.** *Suppose* $\{G_n\}_n$ *is a growing family of graphs with maximum degree* d*, each with sufficiently large expansion as a function of* d *and* δ*. Then the population will converge to consensus on the ground truth with probability* $1 - o(1)$*.*

We believe that max-degree d can be relaxed to a weaker property of sparsity, such as bounded arboricity, in this theorem. For example, in Section 3 we show that under the star topology, the population reaches a consensus on the ground truth with high probability. Yet, the class of max-degree d is *significantly* better understood and technically cleaner to work with. We believe that our analysis of expanders with max-degree d can be leveraged for a better understanding of convergence to consensus in more general classes of graphs, including graphs with alternative sparsity conditions.

1.1 Our Techniques

We prove our main result by dividing an execution of the behavior dynamics into two stages, which we analyze separately. The first stage lasts for a linear (in n, the number of nodes) number of rounds, until most nodes have updated their opinions at least once. We argue that, after the first stage ends, significantly more than half of the individuals (weighted by degree) in the network hold the ground truth as their opinion, with high probability. This argument has two steps: first, we use results from the theory of boolean functions to establish that the expected number of nodes (weighted by degree) with the correct opinion is greater than half of the nodes in the network. Second, to show that the number of correct opinions is concentrated around its expectation, we use the fact that the network has bounded degree. This bounded degree implies that (with high probability) no individual will be very influential after only linearly many steps; indeed, the number of other individuals whose opinions could depend on the private signal of any given node will be small. Hence most pairs of opinions will be independent after linearly many steps, and thus the variance of the number of correct opinions is small.

The second stage begins after most individuals have declared an opinion, and lasts until the dynamics converges. For this stage, we use properties of expander graphs to show that if one opinion has a significant majority in the population, then this bias will be magnified as the process continues, until eventually the entire population reaches consensus. This analysis makes use of the expander mixing lemma as well as the theory of biased random walks. Since the second stage begins in a state where a significant majority of the population (weighted by degree) is reporting the correct opinion (from our analysis of the first stage), we conclude that the population reaches a correct majority with high probability.

While the second half of our argument shares structural similarity with [12], the first half requires a novel approach. Specifically, because [12] studied synchronous learning, they immediately see a correct majority in round one, independent of the graph structure. Due to the asynchronous nature of our learning, showing that we ever reach a correct majority at any point during the process is technically challenging, and requires some assumptions on the graph structure (i. e., sparsity).

Note that we use the two required network properties, expansiveness and sparsity, in different parts of our analysis. The sparsity condition is used to show that opinions are largely independent in the initial rounds of the dynamics, and hence a majority will report correctly. The expansiveness condition is used to show that once the population reaches a clear majority, it will then quickly reach consensus on that majority opinion.

Motivated by this division of the analysis, we make a stronger conjecture that the implications of the two network properties should hold separately. That is, we conjecture that

any network with constant maximum degree leads the population to stabilize in a correct majority. In Appendix A of the full version of the paper, we establish that this is indeed the case under the cycle topology. Furthermore, we conjecture that any network with sufficiently high expansion will stabilize in a consensus (not necessarily a correct consensus).

1.2 Related Work

Our work is related to a line of literature concerning the aggregation of information under Bayesian learning. In the standard learning model, individuals are fully rational and are given noisy signals correlated with a ground truth. The individuals sequentially report a "best guess" at the ground truth. It was first observed by Banerjee [3] and Bikhchandani, Hirshleifer, and Welch [5] that a population may fail to aggregate information when reports are publicly observed, due to information cascades. Smith and Sorensen [13] show that such information cascades can be avoided under the assumption that signals can be arbitrarily informative; i. e., that the strengths of agents' beliefs are unbounded. In a spirit closer to our work, Banerjee and Fudenberg [2] suppose that each agent observes a random subset of the previous agents' actions, and show that asymptotic learning occurs whenever no agent is too influential (i. e., no agent is observed too often). Acemoglu, Dahleh, Lobel, and Ozdaglar [1] show that learning occurs under significantly more general conditions if agents are aware not only of which prior agents they observe, but also the entire history of prior agent observations.

An alternative line of work on social learning concerns the performance of non-Bayesian, heuristic methods of aggregating information. In the classic model of DeGroot [7], each agent's signal is a real number in the unit interval. In each round, agents update their reports by taking a weighted average of their neighbors' reports. Such a process must necessarily converge to a consensus with each connected component of a network. Golub and Jackson [8] consider the question of whether this consensus agrees with the initial ground truth. They find that this occurs if and only if the most influential (i. e., highest-degree) node is vanishingly influential as the population grows large. These models assume a continuous space of opinions and reports. In the case of discrete opinions, where reports are updated by taking the majority report of one's neighbors, Berger [4] shows that it is possible for an initial state with a constant-sized minority to lead ultimately to global adoption of the minority opinion.

The work most similar in spirit to the present paper is Mossel, Neeman, and Tamuz [12]. They consider repeated simultaneous majority dynamics starting from an initial state in which each node takes opinion 0 or 1 independently at random, biased toward 1 (the ground truth). They study conditions under which a majority of the population reports 1 once the dynamics converges; they show that this occurs if the graph is "almost" vertex transitive (in the sense that each vertex can be mapped to many other nodes by graph automorphisms). They also show that if the graph is an expander, then majority dynamics will result in consensus with high probability. Tamuz and Tessler [14] derive sufficient conditions under which the ground truth can be reconstructed from the final state of the dynamics by any means, not necessarily by taking the majority report of the population.

The crucial difference between this line of work and our paper is that they consider synchronous dynamics while the dynamics we consider are asynchronous. One implication of being synchronous is that one might as well assume that all agents start by reporting their signals. (Indeed, if all agents started null, they would switch to reporting their signals on the next step.) To illustrate the significance of this, consider the complete network as an example. If agents all begin by declaring their reports then social learning will almost certainly occur, since the population will immediatley reach consensus on the majority opinion. On the other

hand, if agents begin with null reports and update asynchronously, then the entire population will copy the opinion of the first node that reports and hence there is a good chance that social learning does not occur.

Other lines of work in distributed computation focus on using properties of social networks to show that information can be aggregated efficiently in an algorithmic matter. For example, Kempe Dobra and Gehrke [9] show that gossip-based protocols are particularly successful at aggregating information on networks with good expansion properties.

2 Model and Preliminaries

We consider a social network or graph $G = (V, E)$ with $|V| = n$ individuals. Write $d(v)$ for the degree of v in G, and $Vol(V) = \sum_{v \in V} d(v)$ for the volume of V in G. Individuals live in a world that is in one of two states, say red or blue. Each individual v has a private signal $X(v) \in \{red, blue\}$ regarding the state of the world. These $X(v)$ are conditionally indpendent given the state and are correct with probability $1/2 + \delta$. It will be convenient to assume, without loss of generality, that the state of the world is red and think of $red = 1$ and $blue = 0$. Thus $\Pr[X(v) = 1] = 1/2 + \delta$ for all v.

The individuals stochastically form opinions about the state of the world. Let $C^t(v) \in \{red, blue, uncolored\}$ be the opinion of individual v (or, equivalently, the color of node v) at time t. Initially, individuals hold no opinions and so $C^0(v) = uncolored$. Denote by $N_R^t(v)$ the number of v's neighbors that are colored red at time t, and similarly denote $N_B^t(v)$ the number of v's neighbors that are colored blue at time t. At every time $t > 0$, a node $v \in V$ is chosen uniformly at random. If $N_R^t(v) > N_B^t(v)$, then v is colored red. If $N_R^t(v) < N_B^t(v)$, then v is colored blue. If $N_R^t(v) = N_R^t(v)$, then v is colored $X(v)$.

We first show that for a any graph G, this process stabilizes. That is, with probability 1 there exists a $t < \infty$ such that $C^t(v) = C^{t'}(v)$ for all $t' \geq t$. We do so in a standard way: define a potential function that is initially finite, bounded from below, and decreases by a constant amount in each time step. Intuitively, our potential function counts a combination of the number of bichromatic edges in the graph and the number of self-disagreements, i. e., nodes whose stated opinion differs from their private signal.

▶ **Proposition 1.** *For all G, with probability 1, there exists a t such that $C^t(v) = C^{t'}(v)$ for all $t' \geq t$. Furthermore, the expected number of steps until stabilization is at most $|V|^2 + 2|V||E|$.*

Proof. Define a potential function $F^t(v)$ that is 1 if and only if $C^t(v) \neq X(v)$, and 0 otherwise. Also define a potential function $G^t(e = (u, v))$ that is 2 if either u or v is uncolored, or if $C^t(u) \neq C^t(v)$, and 0 otherwise. Finally, define a potential function $H(t) = \sum_v F^t(v) + \sum_e G^t(e)$. Then $H(0) = |V| + 2|E|$. Furthermore, we claim that if any node's color is changed at time t, then $H(t) < H(t-1)$.

If a node v is the first node in its neighborhood to change from uncolored to colored, then $F^t(v) < F^{t-1}(v)$. Furthermore, $G^{t-1}(e) = 2$ for all e containing v since v was uncolored, so $G^t(e) \leq G^{t-1}(e)$ for all e, and $H(t) < H(t-1)$. If some nodes in v's neighborhood were already colored, then v's color is guaranteed to match the color of at least one neighbor and so for that edge $G^t(e) < G^{t-1}(e)$. For all other edges $G^t(e) \leq G^{t-1}(e)$, and clearly $F^t(v) \leq F^{t-1}(v)$ and so $H(t) < H(t-1)$.

If a node changes colors, then maybe there was a tie among its neighbors. In this case, $\sum_e G^t(e) = \sum_e G^{t-1}(e)$, because we just switch the edges containing v that disagree. But because the color changed with a tie, it must be the case that $F^{t-1}(v) = 1$ and $F^t(v) = 0$. So

again $H(t) < H(t-1)$. Finally, maybe a node changed colors because of a majority among its neighbors. In this case, maybe $F^t(v) = F^{t-1}(v) + 1$, but $\sum_e G^t(e) \leq \sum_e G^{t-1}(e) - 2$ because at least one more edge switches from disagreement to agreement.

Thus, every time a node changes colors (or becomes colored for the first time), the value of H decreases by at least 1, and $H(0) = |V| + 2|E|$, so the process stabilizes after at most $|V| + 2|E|$ changes. If the process has not already stabilized, then there is at least one node that would change colors (or becomes colored for the first time) and it is selected with probability $1/|V|$. So at every step independently there is a color change with probability at least $1/|V|$. Therefore the expected number of steps until a color change is bounded by $|V|$. As the total number of color changes is bounded by $|V| + 2|E|$, the expected number of steps until the process converges is at most $|V|^2 + 2|V||E|$. ◀

It is important to emphasize the distinction between *correct majority* and *consensus*. The former means that more than half of the nodes in the graph are stating the "correct" opinion, while the latter means that every node in the graph is stating the same opinion (not necessarily the correct one).

We conclude this section with formal definitions of sparsity and expansiveness.

▶ **Definition 2.** (Sparsity) There are several different ways to state formally that a graph is sparse. In order from most restrictive to least restrictive, this includes:

- Low fixed degree: The graph is d-regular, and d is small.
- Low maximum degree: Every node in the graph has degree at most d, and d is small.
- Low arboricity: The graph is an edge-union of at most d trees, and d is small.
- Low average degree: The number of edges in the graph is at most dn, and d is small.

Our main result considers the *maximum degree* d notion of sparsity. The example in Section 5.2 of the full version shows that the low average degree notion of sparsity is not restrictive enough to guarantee a correct majority. Our main open question asks whether or not our main result extends to low arboricity as well.

▶ **Definition 3.** (Weighted Adjacency Matrix) The weighted adjacency matrix of a graph G, say $M = M(G)$, is an $n \times n$ matrix defined by

$$M(x,y) = \begin{cases} \frac{1}{\sqrt{d(x)d(y)}} & \text{if } x \text{ and } y \text{ are adjacent in } G, \\ 0 & \text{otherwise.} \end{cases}$$

▶ **Definition 4.** (Expansiveness) A graph G is a λ-expander if all but the first eigenvalue of the weighted adjacency matrix of G lies in $[-\lambda, \lambda]$.

3 Examples

To build intuition for our model and motivate our conjectures, we work through a few specific network topologies in detail before proving our main positive result.

3.1 Complete Graphs

Suppose that G is the complete graph on n vertices. The dynamics proceeds as follows: the node selected in round 1, say v_1, will set $C^1(v_1) = X(v_1)$. That is, v_1 reports its private signal. Every subsequently chosen node will report the majority opinion of the population, and simple induction shows that this will be $X(v_1)$ at all times. The process

will therefore stabilize in a consensus on report $X(v_1)$ with probability 1 for all n. Since $\Pr[X(v_1) = 1] = 1/2 + \delta$, this consensus is correct with probability only $1/2 + \delta$. In other words, the complete graph reaches consensus surely, but exhibits an extreme information cascade in which the population exhibits herding on the first reported signal.

3.2 Star Graphs

We next show that under the star topology, the population will reach a correct consensus with high probability. Suppose G is a star with n leaves. First, we show that the population will certainly reach consensus on the first opinion reported by the center node, say v.

▶ **Claim 1.** *Suppose v is selected by the dynamics for the first time in round t_1. Then, with probability 1, the dynamics reaches consensus on opinion $C^{t_1}(v)$.*

Proof. Suppose $C^{t_1}(v) = R$; the case $C^{t_1}(v) = B$ is handled identically. Then $N_R^{t_1}(v) \geq N_B^{t_1}(v)$, with equality only if $X(v) = R$. For any $t' > t_1$, if a leaf $u \neq v$ is chosen for update, then $C^{t'}(u) = C^{t'}(v)$. That is, node u will copy the opinion of v. Simple induction then shows that, if we write $t_2 > t_1$ for the random variable indicating the round in which v is selected for the second time, we must have $N_R^{t_2}(v) - N_B^{t_2}(v) \geq N_R^{t_1}(v) - N_B^{t_1}(v)$, and hence $C^{t_2}(v) = R$. Applying this argument inductively, we conclude that $C^{t'}(v) = R$ for all $t' > t_1$. Thus each leaf will adopt opinion R each time it is selected for update after time t, and hence the population reaches consensus on R with probability 1. ◀

Write t_1 for the random variable representing the first report time of node v. It remains to show that $C^{t_1}(v) = R$ with high probability. By symmetry, the probability that v chooses an opinion before at least k leaves have chosen opinions is $k/(n+1)$. Conditioning on the event that at least k leaves have reported before t_1, each of their opinions matches their private signals. Applying the additive Chernoff bound, the probability that at most half of them report R at time t_1 is at most

$$\Pr\left[C_R^{t_1} \leq \left(\frac{1}{2} + \delta\right) k - \delta k\right] < e^{-2k\delta^2}$$

Choosing $k = \frac{1}{2\delta^2}\log(n)$ and taking a union bound, we conclude that the probability that at least k leaves are selected before v, and that a majority of those selected leaves take opinion R, is at least $1 - \frac{\log(n)}{2\delta^2 n} - \frac{1}{n} = 1 - o(1)$. We therefore conclude that with probability $1 - o(1)$ the star topology stabilizes in a correct majority.

4 Majority and Consensus

In this section we give a sufficient condition for reaching a correct consensus. More precisely, we focus on a family of λ-expanders of max-degree d and prove that they converge to a correct consensus with high probability.

▶ **Theorem 5.** *Let G be a λ-expander of max-degree d with $\lambda \leq \delta/6$. Then with probability at least $1 - O(\frac{1}{(\delta \ln \ln n)^2})$, the process will terminate in a red consensus.*

Here is a brief outline of our proof. First, we show that in any graph with max-degree d (not necessarily an expander), the volume of nodes with opinion red after $O(n/\delta)$ steps of the process is at least $(1/2 + \delta/2)|E|$ with high probability. We do this by showing that the expected volume of currently red nodes is at least $(1/2 + \delta)|E|$, and then bounding the total pairwise correlation among the colors of nodes to be $o(|E|)$. Combining these two facts with

Chebyshev's inequality gives us the desired claim. Next, we show that for all sufficiently expansive graphs, continuing the stochastic process from a point when the volume of red nodes is at least $(1/2 + \delta/2)|E|$ nodes will result in a red consensus with high probability. Formally, the proof of Theorem 5 follows from Proposition 2 and Corollary 15 after observing that the probability in Corollary 15 is asymptotically dominated by that in Proposition 2.

4.1 Low Degree and Correctness

We would like to count the expected volume of red nodes after a linear number of steps. To this end, we define a Boolean function that specifies the color of a node after a finite sequence of updates. Specifically, let S be any finite sequence of nodes and define a Boolean function f_v^S that takes as input the private signals $\mathcal{X} = \{X(u) \mid \forall u \in V\}$ and outputs the color $C^{|S|}(v)$, when the process chooses nodes in the order specified by S and the private signals are $\mathcal{X}$. If $C^{|S|}(v)$ is uncolored, we define $f_v^S(\mathcal{X})$ to output the private signal $X(v)$; we will later show that this induces a limited degree of overcounting as most nodes are colored after a linear number of steps. Define a random variable

$$f^S(\mathcal{X}) = \sum_v d(v) f_v^S(\mathcal{X})$$

that counts the volume of red nodes after sequence S. Now fix a sequence length T and let f_T be the random variable that selects a sequence S of length T and signals $\mathcal{X}$ at random and outputs $f^S(\mathcal{X})$. Then f_T is the volume of red nodes after T steps of our process. We bound the expectation and variance of f_T and apply Chebyshev to prove that the volume of red nodes is a majority with high probability.

4.1.1 Bounding the Expectation

To bound the expectation, note each f_v^S is *monotone* for all S, v. That is, switching any set of input signals from blue to red can only cause f_v^S to switch from blue to red, but not from red to blue. In addition, f_v^S is *odd* for all S, v. That is, switching all input signals from blue to red and red to blue will cause the output to flip. The following theorem due to Mossel, Neeman, and Tamuz [12], which uses Boolean function analysis, states that such functions on biased random inputs have biased outputs.

▶ **Theorem 6.** *([12]) Let f be an odd, monotone Boolean function. Let $X_1, \ldots, X_n$ be input bits, each sampled i.i.d. from a distribution that is 1 with probability $p \geq 1/2$ and 0 otherwise. Then $\mathbb{E}[f(X_1, \ldots, X_n)] \geq p$.*

The following corollary is a direct application of Theorem 6 and the fact that the private signals $X_i(v)$ are red with probability at least $1/2 + \delta$.

▶ **Corollary 7.** *The expected volume of red nodes at time T, for any T, is at least $(1/2+\delta)|E|$. That is, $\mathbb{E}[f_T] \geq (1/2 + \delta)|E|$ for all T.*

4.1.2 Bounding the Variance

In light of Corollary 7, if we can also bound the variance of f_T, then we can use Chebyshev's inequality to argue that $f_T \geq (1/2 + \delta/2)|E|$ with high probability. Formally, let's define the f^S so that there are n^T separate copies of G, and the private signals $X(v)$ are sampled independently for each copy. Then let f_T be the random variable that picks one S and its corresponding G uniformly at random and outputs f^S. We first state a lemma that allows us to analyze the variance of f_T.

▶ **Lemma 8.** *Let $\{X_1, \ldots, X_n\}$ be random variables all with the same expectation $\mathbb{E}[X_i] = c$, and let X be a random variable that samples from $\{X_1, \ldots, X_n\}$ uniformly at random. Then $Var(X) = \frac{1}{n}\sum_i Var(X_i)$.*

Proof. $Var(X) = \mathbb{E}[X^2] - \mathbb{E}[X]^2 = \mathbb{E}[X^2] - c^2$. $\mathbb{E}[X^2] = \frac{1}{n}\sum_i \mathbb{E}[X_i^2]$. So we get:

$$\begin{aligned} Var(X) &= \frac{1}{n}\sum_i \mathbb{E}[X_i^2] - c^2 \\ &= \frac{1}{n}\sum_i \mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2 \\ &= \frac{1}{n}\sum_i Var(X_i). \end{aligned}$$

◀

To use Lemma 8, we need to modify our random variables slightly so that they all have the same expectation. To do this, just define $g^S = f^S - (\mathbb{E}[f^S] - \frac{1+2\delta}{2}|E|)$, and g_T to sample S uniformly at random and then sample g^S. By Corollary 7, $f^S \geq g^S$ for all S always. Therefore, showing that $g_T \geq (1/2 + \delta/2)|E|$ with high probability suffices to prove that $f_T \geq (1/2 + \delta/2)|E|$ as well.

So now let's analyze the variance of g_T. Lemma 8 tells us that the variance of g_T is just the average of the variances of each $Var(g^S)$. Furthermore, we can write the variance of each g^S as

$$Var(g^S) = \sum_{u,v} d(u)d(v)Cov(f_u^S, f_v^S)$$

and therefore, we can write $Var(g_T)$ as

$$Var(g_T) = \frac{1}{n^T}\sum_S \sum_{u,v} d(u)d(v)Cov(f_u^S, f_v^S).$$

Now we observe that $Var(g_T)$ is exactly the expected value of the following random process: sample two nodes u and v uniformly at random (with replacement), sample a sequence of length T uniformly at random, and compute $n^2 d(u)d(v)Cov(f_u^S, f_v^S)$. Furthermore, as each f_v^S is a 0-1 random variable, $Cov(f_u^S, f_v^S) \leq 1$. As $Cov(f_u^S, f_v^S) = 0$ when f_u^S and f_v^S are independent, we can define G_T to be a random variable that is 0 whenever S, u, v are sampled such that f_u^S and f_v^S are independent and 1 otherwise. The reasoning above shows that if we show that $\mathbb{E}[G_T] \leq c$, then $Var(g_T) \leq cd^2n^2$.

So now our aim is to study G_T. Let's first ask what private signals can possibly affect the color of node v at the end of sequence S. If t_v is the last step that v is chosen to update its color, then f_v^S is clearly a function of the colors of v's neighbors at time t_v. Furthermore, if we look at any neighbor u of v, and let t_u be the last step that u is chosen to update its color before t_v, then the color of u at time t, is clearly a function of the colors of u's neighbors at time t_u (as $t \geq t_u$, and node u does not update its color between t and t_u). Iterating this reasoning out, we can define the set $N^S(v)$ to be those nodes u such that there is a path $v, x_1, \ldots, x_k, u$ from u to v and corresponding times $t_v > t_1 > \ldots > t_k > t_u$ such that u announces its color at time t_u in S, v announces its color at time t_v in S, and each x_i announces its color at time t_i in S. We then see that f_v^S can be written as a function of only the signals $\{X(u)\}_{u \in N^S(v)}$. Therefore, if $N^S(v) \cap N^S(u) = \emptyset$, it is necessarily the case that

f_u^S and f_v^S are independent, as they are functions on disjoint sets of independent random variables. So our approach to bounding $\mathbb{E}[G_T]$ will be to analyze the probability that when v and u are chosen uniformly at random (with replacement) and S is a random sequence of length T that $N^S(v) \cap N^S(u) = \emptyset$.

We do this by studying the random variable $|N^S(v)|$ for a random node v and random sequence S. We can compute $N^S(v)$ by initializing $N^S(v) = \emptyset$ and tracking backwards through S. Until the first (moving backwards in time) time that v announces its color, $N^S(v) = \emptyset$. When v first updates its color, we update $N^S(v) = \{v\}$. From here, until the next time that a neighbor of v announces its color, $N^S(v)$ remains unchanged. When the first neighbor u of v updates its color, we update $N^S(v) = \{v, u\}$. Iterating this reasoning, we can compute $N^S(v)$ by tracking backwards through S, updating $N^S(v)$ to $\{v\}$ the first time that v announces its color, and then updating $N^S(v) := N^S(v) \cup \{u\}$ any time a neighbor u of $N^S(v)$ announces its color.

So let N_i be the random variable denoting the number of steps between when $|N^S(v)|$ first becomes $i-1$ and when $|N^S(v)|$ first becomes i over the random choice of S. Recall S is chosen uniformly at random from all sequences of length T. As each node has degree at most d, and the neighborhood $N^S(v)$ is a connected subgraph, when $|N^S(v)| = i - 1 \geq 2$, there are at most $(i-1)(d-1)$ ways to grow $N^S(v)$ (and for $i - 1 = 1$, there are at most d ways). Thus the N_i are independent geometric random variables with mean at least $\frac{n}{1+(i-1)(d-1)}$. For ease of analysis, we analyze each N_i as independent random variables of mean exactly $\frac{n}{id}$ (this is valid because these random variables are stochastically dominated by the actual N_i, meaning that we are only underestimating the number of steps needed for $|N^S(v)|$ to grow). Now we see that, for any x, if we define N^x to be the number of steps before $|N^S(v)| = x$, then N^x is exactly $\sum_{i=1}^{x} N_i$. As each N_i is a geometric random variable with parameter id/n, $\mathbb{E}[N_i] = \frac{n}{di}$, and $Var(N_i) = \frac{n^2}{i^2 d^2}$. So because all N_i are independent, we get that:

$$\mathbb{E}[N^x] = \sum_{i=1}^{x} \frac{n}{di} \geq \frac{n \ln x}{d}, \quad Var(N^x) = \sum_{i=1}^{x} \frac{n^2}{i^2 d^2} = \frac{\pi^2 n^2}{6d^2},$$

$$\sigma(N^x) = \sqrt{Var(N^x)} \leq \frac{2n}{d}.$$

So by Chebyshev's inequality, we get that $Pr[N^x \leq \frac{n \ln x}{d} - t\frac{2n}{d}] \leq \frac{1}{t^2}$, which can be rewritten as:

$$Pr\left[N^x \leq (1-\epsilon)\frac{n \ln x}{d}\right] \leq \frac{4}{(\epsilon \ln x)^2}. \tag{1}$$

From here, we simply observe that if the shortest path from u to v has length $> 2x$, and $|N^S(v)|, |N^S(u)| \leq x$, then $N^S(u) \cap N^S(v) = \emptyset$. We also observe that the number of nodes within distance $2x$ of v is bounded by d^{2x} for all x. So when u and v are chosen uniformly at random (with replacement) we have:

$$Pr[dist(u, v) \leq 2x] \leq \frac{d^{2x}}{n}.$$

Taking $T = \frac{n \ln x}{2d}$ corresponds to setting $\epsilon = 1/2$ in Equation (1). So for any u, v, the union bound guarantees that with probability at most $\frac{32}{(\ln x)^2}$ $|N^S(v)|, |N^S(u)| \geq x$. Furthermore, if u, v, S are chosen uniformly at random, we see that with probability at most $\frac{d^{2x}}{n}$, $dist(u, v) \leq 2x$. Again taking a union bound, the probability that either of these events occur is at most $\frac{32}{(\ln x)^2} + \frac{d^{2x}}{n}$. And in the event that none of these events occur, we clearly have $N^S(u) \cap N^S(v) = \emptyset$. Therefore, we conclude that for all x, if $T = \frac{n \ln x}{2d}$, $\mathbb{E}[G_T] \leq \frac{32}{(\ln x)^2} + \frac{d^{2x}}{n}$.

By the reasoning above, we have now shown that when $T = \frac{n \ln x}{2d}$, we have:

$$Var(g_T) \leq d^2 n^2 \left(\frac{32}{(\ln x)^2} + \frac{d^{2x}}{n} \right) .$$

To simplify notation, we observe that whenever $x = o(\log n)$ the first term asymptotically dominates the second. So we will restrict ourselves to setting $x = o(\log n)$ and rewrite:

$$Var(g_T) \leq \frac{33 d^2 n^2}{(\ln x)^2}$$

So we can apply Chebyshev's inequality to g_T now and see that whenever $x = o(\log n)$, we have:

$$Pr\left[g_T \leq (1/2 + \delta)|E| - t \cdot \frac{8dn}{\ln x} \right] \leq \frac{1}{t^2} .$$

And plugging in for $t = \delta \ln(x)/(32d)$ we get:

$$Pr[g_T \leq (1/2 + 3\delta/4)|E|] \leq \frac{1024 d^2}{(\delta \ln x)^2} .$$

And because $f_T \geq g_T$ always, we have:

$$Pr[f_T \leq (1/2 + 3\delta/4)|E|] \leq \frac{1024 d^2}{(\delta \ln x)^2} .$$

Finally, recall that in order to make f_T odd, we had to define f_v^S to be $X(v)$ in the event that v does not announce its color at all in S. So f_T does not exactly count the number of red nodes because its getting credit for some nodes with a red private signal who haven't actually announced a color at all. But this is easy to cope with: we can just show that with high probability the volume of nodes that have yet to announce a color after $\frac{n \ln x}{2d}$ steps is at most $\delta|E|/4$. Note that because all nodes have degree at most d, it is sufficient to show that the *number* of nodes who have yet to announce a color is at most $\delta n/(4d) \leq \delta|E|/(4d)$ with high probability.

For a single node v, the probability that v has not yet announced a color after $\frac{n \ln x}{2d}$ is exactly:

$$(1 - 1/n)^{\frac{n \ln x}{2d}} \leq e^{-\frac{\ln x}{2d}} \leq x^{-\frac{1}{2d}} .$$

So if we define $C_x(v)$ to be the indicator random variable that is 1 if v has not yet announced a color by time $\frac{n \ln x}{2d}$, and 0 otherwise, the collection of random variables $\{C_x(v)\}_v$ are negatively correlated. So if we define $C_x = \sum_v C_x(v)$, we get $\mathbb{E}[C_x] = nx^{-\frac{1}{2d}}$. Using the additive Chernoff bound, we get:

$$Pr\left[C_x \geq nx^{-\frac{1}{2d}} + tn \right] \leq e^{-2t^2 n} .$$

And plugging in for $t = \delta/(4d) - x^{-\frac{1}{2d}}$ we get:

$$Pr[C_x \geq \delta n/(4d)] \leq e^{-n(\delta/(4d) - x^{-\frac{1}{2d}})^2} .$$

Because δ and d are constant and $x = o(\log n)$,[2] this is clearly asymptotically dominated by $\frac{1}{(\delta \ln x)^2}$. So taking a union bound over the probability that more than $\delta/4$ nodes have yet to announce a color and the probability that $f_T \leq (1/2 + 3\delta/4)n$, we get the following proposition:

[2] In fact, this would still be true if we took $x = O((\log n)^{1-\epsilon})$ for some $\epsilon > 0$, $1/\delta = O(x)$, and $d = o(\frac{\ln x}{\ln(1/\delta)})$

▶ **Proposition 2.** *For any $x = o(\log n)$ and $T = \frac{n \ln x}{2d}$:*

$$Pr[\textit{volume of announced reds at time } T \leq (1/2 + \delta/2)|E|] \leq \frac{1025}{(\delta \ln x)^2}\,.$$

In particular, when $x = \ln \ln n$ and $T = \frac{n \ln \ln \ln n}{2d}$, this probability is at most $O\left(\frac{1}{(\delta \ln \ln n)^2}\right)$.

4.2 Expansion and Consensus

In this section, we apply a different argument based on expansion to show that if G is sufficiently expansive, once the volume of nodes that have announced red exceeds $(1/2 + \delta/2)|E|$, it is extremely likely that the process will continue to stabilize in a red consensus. This argument has two steps. First, we apply an argument of [12] to show that, in an expansive network, the volume of nodes that will switch from blue to red if chosen is a constant factor larger than those that would switch from red to blue if chosen, conditioned on the fact that the volume of nodes announcing red is at least $(1/2 + \delta/4)|E|$. Second, we argue that with very high probability, due to this fact, if the volume of nodes announcing red starts above $(1/2 + \delta/2)|E|$, then we will reach the point where all nodes have announced red before we reach a point where the volume of nodes announcing red is only $(1/2 + \delta/4)|E|$. This second step proceeds by coupling the convergence process to an absorbing random walk, and applying the theory of biased random walks.

In the following lemmas, let R denote the set of nodes who have currently announced red, and B the set of nodes who have currently announced blue or nothing. Let also R' denote the set of nodes that would announce red if they were chosen, and B' the set of nodes that would announce blue if they were chosen.

The following lemma relates the number of edges between two sets of nodes in an expander with max-degree d to their expected number in a random graph.

▶ **Lemma 9.** *([6]) If G is a λ-expander of max-degree d, then for any two subsets $S, T \subseteq V$, let $E(S,T)$ denote the number of edges between S and T (double-counting edges from $S \cap T$ to itself). Then:*

$$|E(S,T) - \frac{Vol(S)Vol(T)}{|E|}| \leq \lambda\sqrt{Vol(S)Vol(T)}\,.$$

Using Lemma 9, we can bound the number of "potential" B nodes.

▶ **Corollary 10.** *If G is a λ-expander of max-degree d with $\lambda \leq \frac{\delta}{6}$ and $|R| \geq (1/2 + \delta/4)n$, then $|B'| \leq |B|/2$.*

Proof. We know that every node in B' has at least half of its neighbors in B (or else they would choose red). Therefore, $E(B', B) \geq Vol(B')/2$. In addition, Lemma 9 tells us that $E(B', B) \leq \frac{Vol(B')Vol(B)}{|E|} + \lambda\sqrt{Vol(B)Vol(B')}$. Putting these two together, we get:

$$Vol(B')/2 \leq \frac{Vol(B')Vol(B)}{|E|} + \lambda\sqrt{Vol(B')Vol(B)}.$$

Reorganizing the last inequality we get

$$Vol(B') \leq Vol(B)\left(\frac{\lambda}{\frac{1}{2} - \frac{Vol(B)}{|E|}}\right)^2\,.$$

Applying the fact that $Vol(B)/|E| \leq 1/2 - \delta/4$ we get

$$Vol(B') \leq Vol(B)\left(\frac{16\lambda^2}{\delta^2}\right).$$

Finally, by the fact that $\lambda \leq \frac{\delta}{6}$ we get

$$Vol(B') \leq Vol(B)\left(\frac{16\delta^2}{36\delta^2}\right) \leq Vol(B)/2,$$

as desired. ◀

Now, we make use of Corollary 10 to show that we are very likely to switch more blues to reds than reds to blues over many announcements.

▶ **Corollary 11.** *If $Vol(B') \leq Vol(B)/c$, then $Vol(B \cap R') \geq cVol(R \cap B')$, and $Vol(B \cap R') \geq 1$. In other words, the volume of nodes who will switch from blue to red is at least c times the number of nodes who will switch from red to blue if chosen, and there is at least 1 such node.*

Proof. We know that $Vol(B \cap B') = x$, for some $x \geq 0$. So we can write $Vol(B \cap R') = Vol(B) - x$ and $Vol(B' \cap R) = Vol(B') - x$. Combining this with the fact that $Vol(B) \geq cVol(B')$ we get:

$$\frac{Vol(B \cap R')}{Vol(R \cap B')} \geq \frac{cVol(B') - x}{Vol(B') - x}.$$

Because $x \geq 0$, this is always at least c. As $Vol(B') < Vol(B)$, there must be at least one node in $B \cap R'$. ◀

To complete our analysis, we use the theory of biased random walks.

▶ **Definition 12.** For $d \geq 1$ and $p > 0$, a d-bounded, p-biased random walk on the integers is a sequence $(Z_t)_{t \geq 0}$ such that:

- $Z_0 = 0$,
- Z_t depends only on $(Z_0, \ldots, Z_{t-1})$,
- $|Z_t - Z_{t-1}| \leq d$ for each $t \geq 1$, and
- for all $(Z_t)_{t<T}$, $\mathbb{E}[Z_T \mid Z_0, \ldots, Z_{T-1}] \geq Z_{T-1} + p$.

The following lemma establishes a crucial property of biased random walks, which is then used in the remainder of this section to show that once the volume of red nodes reaches a certain threshold, the process will converge to a red consensus with high probability.

▶ **Lemma 13.** *Let $(Z_t)_{t \geq 0}$ be a d-bounded p-biased random walk on the integers. Then, for any $x > 0$, the probability that the walk reaches a value less than $-x$ before a value greater than x is at most $\frac{2x}{p} e^{-px/4d^2}$.*

Proof. For each $t \geq 1$, define $Y_t = Z_t - Z_{t-1}$, and let $W_t = Y_t - \mathbb{E}[Y_t \mid Y_1, \ldots, Y_{t-1}]$. Note that the sequence $(W_t)_{t \geq 1}$ forms a martingale, whose entries lie in $[-d, d]$. The Azuma-Hoeffding inequality then implies that, for any $n \geq 1$,

$$\Pr\left[\sum_{t=1}^{n} W_t < -x\right] \leq e^{-x^2/2nd^2}.$$

Let A_n be the event that there exists any prefix of the sequence $(W_t)_{t \leq n}$ with sum less than $-x$. Taking a union bound over all t between 1 and n, we have that the probability of event A_n occurring is at most $n \cdot e^{-x^2/2nd^2}$.

If we condition on A_n not occurring, then observe that for each $T \leq n$,

$$Z_T = \sum_{t=1}^{T} Y_t = \sum_{t=1}^{T} W_t + \mathbb{E}[Y_t \mid (Y_k)_{k<t}] > \mathbb{E}[Z_T] - x.$$

In particular, $Z_n > \mathbb{E}[Z_n] - x$ and moreover $Z_t > -x$ for all $t \leq n$. If we choose $n = 2x/p$, then $\mathbb{E}[Z_n] > pn = 2x$, and hence $A_{2x/p}$ not occurring implies that $Z_n > x$ and $Z_t > -x$ for all $t < n$, as required. Furthermore, the probability of $A_{2x/p}$ is at most $\frac{2x}{p} \cdot e^{-px/4d^2}$. ◀

We now apply Lemma 13 to the stochastic process, letting Z_t be the volume of red nodes. The hypotheses of Corollary 14 below (and the fact that G has maximum degree d) guarantee that the random walk is $\frac{c-1}{c+1}$-biased and d-bounded.

▶ **Corollary 14.** *Let R_0 and B_0 be such that $Vol(R' \cap B_0) \geq cVol(B' \cap R_0)$. For any x, if R and B maintain this property whenever $Vol(B_0) - x \leq Vol(B) \leq Vol(B_0) + x$ (and therefore $Vol(R_0) - x \leq Vol(R) \leq Vol(R_0) + x$ as well), then the probability that we arrive at a state with $Vol(B) \geq Vol(B_0) + x$ before one with $Vol(R) \geq Vol(R_0) + x$ is at most $2x(\frac{c+1}{c-1})e^{-(c-1)x/4(c+1)d^2}$.*

Proof. Corollarycor:randomwalk Consider a biased one-dimensional random walk that takes ℓ steps up whenever a node of degree ℓ switches from blue to red, and ℓ steps down whenever a node of degree ℓ switches from red to blue. Then the corollary is exactly studying the probability that this random walk reaches a depth of $-x$ before a height of x.

This walk is d-bounded. We also claim that it is $(\frac{c-1}{c+1})$-biased. To see this, let W^+ be the expected upward step of the walk on a given round; i. e., the expected step of the walk if we were to replace any negative movement by 0. Likewise, let $W^- \leq 0$ be the expected downward step. Note then that the expected step is $W^+ + W^-$. Since $Vol(R' \cap B_0) \geq cVol(B' \cap R_0)$, we have $W^+ \geq cW^-$. Also, $W^+ - W^- \geq 1$, since each step is of distance at least 1. We can then conclude that $W^+ + W^- \geq \frac{c-1}{c+1}(W^+ - W^-) = \frac{c-1}{c+1}$. Now, by Lemma 13, the probability that this walk reaches depth $-x$ first is at most $2x(\frac{c+1}{c-1})e^{-(c-1)x/4(c+1)d^2}$. ◀

Finally, we use Corollary 14 to prove that the stochastic process terminates in a consensus. The idea is that once we have reached $Vol(R) \geq (1/2 + \delta/2)|E|$, the expansiveness of G guarantees that the hypotheses of Corollary 14 are satisfied. We then iteratively apply Corollary 14 to show that we are extremely likely to reach a state with $Vol(R) \geq (1/2 + k\delta/2)|E|$ before we reach a state with $Vol(R) \leq |E|/2$, for all integers $k \in [2/\delta]$.

▶ **Corollary 15.** *If G is a λ-expander with max-degree d and with $\lambda \leq \frac{\delta}{6}$, and the stochastic process reaches a point where $Vol(R) \geq (1/2 + \delta/2)|E|$, then with probability at least $1 - 4n \cdot e^{-\delta n/48d^2}$, the process will terminate in a red consensus.*

Proof. Once the process reaches a point where $Vol(R) \geq (1/2 + \delta/2)|E|$, we will have $Vol(R) \geq (1/2 + \delta/4)|E|$ until the volume of reds that switch to blue is at least $\delta|E|/4$ more than the volume of blues that switch to red. Therefore, by Corollaries 10, 11, and 14, the probability that we reach a point where $Vol(R) = (1/2 + \delta/4)|E|$ before we reach a point where $Vol(R) = (1/2 + 3\delta/4)|E|$ is at most

$$6(\delta n/4)e^{-(\delta n/4)/12d^2} < 2\delta n e^{-\delta n/48d^2}.$$

Similarly, once we have reached a point where $Vol(R) = (1/2 + i\delta/4)|E|$ $(2 \leq i < 2/\delta)$, the probability that we reach a point where $Vol(R) = (1/2 + (i-1)\delta/4)|E|$ before we reach a point where $Vol(R) = (1/2 + (i+1)\delta/4)|E|$ is at most $2\delta n e^{-\delta n/48d^2}$. Therefore, we can take

a union bound over all $2 \leq i < 2/\delta$ and say that with probability at least $1 - 4ne^{-\delta n/48d^2}$, the first time we hit $Vol(R) = (1/2 + i\delta/4)|E|$, we will hit $Vol(R) = (1/2 + (i+1)\delta/4)|E|$ before we hit $Vol(R) = (1/2 + (i-1)\delta/4)|E|$, for all $2 \leq i < 2/\delta$. In the event that this happens, we will hit a red consensus before we hit $Vol(R) = (1/2 + \delta/4)|E|$, and therefore the process will stabilize in a red consensus. ◀

5 Conclusion

We study whether information aggregates efficiently under natural dynamics in social networks with "real-world" properties. We show that if each individual's signal agrees with the ground truth with probability at least $1/2 + \delta$, independently, then the entire society is likely to agree on the ground truth with high probability (approaching 1 as $n \to \infty$) in the class of λ-expanders with maximum degree d for any fixed $d, \lambda \leq \frac{\delta}{6}$. We also analyze separately the example of a star on n nodes, and show that it also achieves a consensus on the ground truth with high probability. This suggests that our results apply to additional notions of sparsity. An interesting direction for future work would be to show that more general classes of "sparse" expanders reach consensus on the ground truth with high probability. One possibility is the set of expanders with arboricity of at most d. Additionally, the use of sparsity and expansiveness is decoupled in our analysis: sparsity is used to show that a correct majority is reached at some point during the process, and expansiveness is used to show that, once this occurred, the process terminates in a correct consensus. These results suggest two interesting directions for future research. First, we conjecture that sparsity (e. g., low arboricity) guarantees that the process stabilizes in a correct majority, as in the ring. Second, we showed that expansiveness guarantees that once enough of a (possibly incorrect) majority forms, the process terminates in a consensus with high probability. We conjecture that all expansive graphs terminate in a (possibly incorrect) consensus with high probability.

References

1 Daron Acemoglu, Munther A. Dahleh, Ilan Lobel, and Asuman Ozdaglar. Bayesian learning in social networks. *Review of Economic Studies*, 78(4):1201–1236, 2011.

2 Abhijit Banerjee and Drew Fudenberg. Word-of-mouth learning. *Games and Economic Behavior*, 46(1):1–22, January 2004.

3 Abhijit V. Banerjee. A simple model of herd behavior. *The Quarterly Journal of Economics*, 107(3):797–817, 1992.

4 Eli Berger. Dynamic monopolies of constant size. *J. Comb. Theory, Ser. B*, 83(2):191–200, 2001.

5 Sushil Bikhchandani, David Hirshleifer, and Ivo Welch. A theory of fads, fashion, custom, and cultural change in informational cascades. *Journal of Political Economy*, 100(5):992–1026, October 1992.

6 F. Chung and R. Graham. Quasi-random graphs with given degree sequences. *Random Structures & Algorithms*, 32(1):1–19, 2008.

7 Morris H. DeGroot. Reaching a consensus. *Review of Economic Studies*, 69(345):118–121, 1974.

8 Benjamin Golub and Matthew O. Jackson. Naïve learning in social networks and the wisdom of crowds. *American Economic Journal: Microeconomics*, 2(1):112–149, 2010.

9 David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *FOCS*, pages 482–491, 2003.

10 Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW*, pages 695–704, 2008.

11 Fragkiskos D. Malliaros and Vasileios Megalooikonomou. Expansion properties of large social graphs. In *DASFAA Workshops*, pages 311–322, 2011.

12 Elchanan Mossel, Joe Neeman, and Omer Tamuz. Majority dynamics and aggregation of information in social networks. In *Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2013.

13 Lones Smith and Peter Sorensen. Pathological outcomes of observational learning. *Econometrica*, 68(2):371–398, March 2000.

14 Omer Tamuz and Ran Tessler. Majority dynamics and the retention of information. In *Working paper*, 2013.

Deliver or Hold: Approximation Algorithms for the Periodic Inventory Routing Problem

Takuro Fukunaga[1], Afshin Nikzad[2], and R. Ravi[3]

1 **National Institute of Informatics, Japan**
JST, ERATO, Kawarabayashi Large Graph Project, Japan
takuro@nii.ac.jp
2 **MS&E Department, Stanford University, USA**
nikzad@stanford.edu
3 **Tepper School of Business, Carnegie Mellon University, USA**
ravi@cmu.edu

Abstract

The inventory routing problem involves trading off inventory holding costs at client locations with vehicle routing costs to deliver frequently from a single central depot to meet deterministic client demands over a finite planing horizon. In this paper, we consider periodic solutions that visit clients in one of several specified frequencies, and focus on the case when the frequencies of visiting nodes are nested. We give the first constant-factor approximation algorithms for designing optimum nested periodic schedules for the problem with no limit on vehicle capacities by simple reductions to prize-collecting network design problems. For instance, we present a 2.55-approximation algorithm for the minimum-cost nested periodic schedule where the vehicle routes are modeled as minimum Steiner trees. We also show a general reduction from the capacitated problem where all vehicles have the same capacity to the uncapacitated version with a slight loss in performance. This reduction gives a 4.55-approximation for the capacitated problem. In addition, we prove several structural results relating the values of optimal policies of various types.

1998 ACM Subject Classification G.1.6 Optimization

Keywords and phrases Inventry Routing Problem, Approximation algorithm, Prize-collecting Steiner Tree

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.209

1 Introduction

The Inventory Routing Problem (IRP) is a classical problem in the Supply Chain Optimization area of the Operations Management literature [7, 13, 18, 19, 21] that captures the trade-off between the holding costs for inventory and the routing costs of replenishing the inventory at various locations in a supply chain. It arises in the context of vendor-managed inventory systems where the supplier running a depot manages the inventory at its client demand locations [30]. The general problem involves multiple products that are stocked at multiple depots, that must be shipped to meet the demand for these products arising at multiple locations (clients) specified over the course of a planning horizon that involves several time periods (days or rounds). The costs of holding a unit of each product per day at each of the clients are specified to compute the inventory holding costs; vehicles are available at the depots with given capacities and transportation costs in the metric defined by the depots and clients determine the vehicle routing costs. The goal of the problem is to find a set of vehicle

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 209–225

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

routes for each day of the planning horizon that delivers sufficient units of products at each location to satisfy all demands, and minimizes the total sum of inventory *holding* costs and vehicle *routing* costs over the whole planning horizon. Note that we are not allowed to have any demand backlog at any client location.

Being a natural extension of the classical inventory holding problem to a networked setting, many of the early papers try to adopt the approach from the inventory literature of looking for *policies* under a specified random pattern of demand at the locations [2, 3] that optimize the infinite-horizon costs. However, no optimal or near-optimal policies are known in the general case of the problem. The more tactical solutions to this real-life problem involve focusing on a finite planning horizon as we have formulated it above: in this given horizon, the demands can be assumed to be deterministic (realizations of some underlying random process or very good forecasts of them). The problem can then be cast as an integer program [14] that decides on the vehicle routes per day and the amounts delivered per day. Solution approaches for this formulation typically involve heuristics [15, 17] that try to manage the short-term formulation with extra constraints or objectives to take care of extending the solutions to the next finite planning horizon overlapping with the current one.

In this paper, we address the most-studied version of the finite horizon problem with a *single depot*. We focus on the tactical version of the problem where the demands at the clients are assumed to be deterministic over the horizon and design constant-factor approximation algorithms for periodic solutions which we define next.

Periodic Schedules

Even though the single-depot version of the IRP can be cast as an IP, finding solutions that serve the clients at arbitrarily spaced time periods can be practically cumbersome to implement. Indeed, Anily and Federgruen noted early on [2]: "The complexity of the structure of optimal policies makes them difficult, if not impossible, to implement even if they could be computed efficiently". A very natural restriction that has been considered is to require that every client is visited according to a *periodic* schedule, i.e., once every f days for some frequency f associated with that client. A simple example is a delivery schedule that visits a client in a particular day of the week.

Another common further restriction on the set of frequencies used is that longer frequencies are multiples of all the smaller frequencies: e.g., clients can be visited once every week, or once every month (4 weeks) or once every quarter (every 12 weeks), so that in days, the periods are 7, 28 and 84. Such schedules are called *nested periodic* [31]. A particular class of such policies where the frequencies are powers of two is very well studied in Inventory Management following the seminal work of Roundy [27] on the efficacy of power-of-two policies as very good approximations for more complicated replenishment policies. Our main result is a constant-factor approximation algorithm for nested periodic schedules, that also gives corollaries for power-of-two schedules as well as for general power-of-k schedules for $k > 2$ (We call these 2-periodic and k-periodic schedules in the sequel).

Partition Schedules

While we require each node to be visited with a frequency that is a power of two in the 2-periodic schedules, we have a choice of the vehicle route used in round 2 versus round 4. In the general problem, we may use two different vehicle routes on days 2 and 4 since the second route will also include nodes visited with frequency 4 in addition to those visited with frequency 2: we call such schedules simply 2-periodic schedules to denote their generality.

However, we may further restrict that the vehicle route used to visit the clients of frequency 2 must always be the same; in this case, in round 4, we would have two different vehicle routes, with one visiting all nodes of frequency 2 and the other visiting nodes of frequency 4. Note that we have effectively partitioned the clients into subsets based on their frequencies and the schedule specifies a vehicle route that visits each set independent of the others. Such schedules are called *partition schedules* or policies [3]; we will term a partition schedule for the general nested case as a nested partition policy and the power-of-two version as a 2-partition policy. Note that partition schedules are also periodic.

2 Our Results

2.1 Problem Definition

In the IRP, we are given a complete graph G with a node set $V(G)$ and an edge set $E(G)$, and a metric distance function w (i.e., $w(ab) + w(bc) \geq w(ac)$ for any nodes $a, b, c \in V(G)$). A single depot node (or root) r is specified as well as a set of client nodes. The depot has infinite production capacity, which means there are as many units of the production as we need available at the depot.

We are given a planning horizon of length $T + 1$ (our rounds are numbered 0 through T) and are asked to satisfy the demands of the clients in these $T + 1$ rounds. Thus, client i, which has a demand d_i^j in the j-th round of the planning horizon, should be satisfied by the depot using a set of (possibly capacitated) vehicles as follows: In each round, for each vehicle, we determine a sequence of some nodes of $V(G)$, and the vehicle visits those nodes in that order. The vehicle starts at the depot node, picks up the products, and then visits client nodes, to deliver a portion of its shipment to the client and satisfy the client's demands until the next visit; in the capacitated version, the vehicle cannot carry more units than its capacity C. In this paper, we assume uniform capacities for all the vehicles. Also, we assume that the client demands are discrete, and that partial (fractional) satisfaction of a unit of demand is not allowed.

Assume inventory is used in the order it was delivered to satisfy demand. For each unit of the production that client i receives in round s and keeps in its inventory till round t, it incurs a holding cost $h_{s,t}^i$. Hence the client i is visited in rounds $0, t_1, \ldots, t_k$, then the holding cost for i is $\sum_{j=0}^{k} \sum_{s=t_j}^{t_{j+1}} d_i^s h_{t_j s}^i$, where we let $t_0 = 0$ and $t_{k+1} = T + 1$. Note the general structure used for modeling the holding costs. The only assumption we need for the holding costs is monotonicity, i.e. for any client i and any three rounds $u \leq s \leq t$, we need to have $h_{ut}^i \geq h_{st}^i$.

For a given solution x, the sum of all the holding costs incurred by all the clients over the whole horizon is called the *holding cost* $h(x)$. Similarly, the sum of the distances traveled by all the vehicles in all the rounds is called the *shipping cost* $w(x)$. The cost of a solution x to the IRP, $c(x)$, is the sum of its holding cost and its shipping cost. We want to determine the solution (also called schedule or policy), that specifies vehicle routes and demand deliveries in each route, such that the total incurred cost is minimized. Note that for feasibility, the amount at any location at any period is at least its demand up to now.

As noted above, we consider only periodic policies in this paper. A frequency is defined as a positive integer, and we assume that available frequencies are given. In periodic policies, a solution assigns a frequency to each client node. If a client node is assigned frequency f, then the vehicle in round pf has to visit the node for each positive integer p up to $\lfloor T/f \rfloor$. Partition policies are different from periodic policies only in the fact that a client node is visited in the same route every time.

Let $q_0, q_1, \ldots, q_k$ be integers such that $q_0 = 1$ and $q_j \geq 2$ for $j = 1, 2, \ldots, k$. In a nested (periodic or partition) policy, the available frequencies are $f_0, f_1, \ldots, f_{k+1}$ that are defined as $f_i = \prod_{j=0}^{i} q_j$ for $i = 0, 1, \ldots, k$, and $f_{k+1} = +\infty$. We need f_{k+1} to represent nodes visited only once in round 0. A k'-periodic policy denotes a policy with frequencies defined by $q_j = k'$ for each $j = 1, 2, \ldots, k$.

Summing up, the problem considered in this paper is summarized as follows.

Problem IRP

Input:

- A metric (G, w) with a depot node $r \in V(G)$.
- A planning horizon length $T + 1$.
- A demand d_i^j for each $i \in V(G) \setminus \{r\}$ and $j \in \{0, 1, \ldots, T\}$.
- A holding cost $h_{s,t}^i$ for each $i \in V(G) \setminus \{r\}$ and $s, t \in \{0, 1, \ldots, T\}$.
- A vehicle capacity C in capacitated case.
- Available frequencies $f_0, f_1, \ldots, f_{k+1}$, where $f_{k+1} = +\infty$. The nested policy defines f_i from $q_0, q_1, \ldots, q_k$ as $\prod_{j=0}^{i} q_j$ for $i = 0, 1, \ldots, k$.

Output:

- Allocation of a frequency to each client.
- A set of vehicle routes for each round.

Objective: Minimize the sum of the holding cost and the shipping cost.

2.2 Techniques

We present constant-factor approximation algorithms for designing periodic policies for the general capacitated version of the IRP. Our main contribution is a simple reduction of the problem to a carefully designed instance of a prize-collecting vehicle routing problem (PCVRP), where vehicles from a root node (depot) must either visit each client or pay a pre-specified penalty for not visiting it, with the objective function summing the cost of the route and the penalties of uncovered clients. The prize-collecting TSP [6] as well as the prize-collecting Steiner tree problems have been well studied [10, 22]; the best-known performance ratios for these problems are some constants slightly below 2 [4].

We describe the key idea behind our algorithm for the uncapacitated case where the vehicles can carry any number of units in each round: our reduction uses roughly one copy of the metric for each frequency, where the edge lengths in the metric are scaled appropriately by the number of times the vehicle route of the corresponding frequency will be used in the overall IRP schedule: if the planning horizon is T rounds, and the frequency represented by the i^{th} copy is f_i, this scale factor is roughly T/f_i. The penalties of a node in each of the copies are carefully set so that if a node is covered by a vehicle route for the first time in the i^{th} copy, then the sum of the penalties of the first $(i-1)$ copies gives the holding cost for this node if it were visited only every f_i rounds. In this way, the two components of the PCVRP problem - covering with routes and penalties - capture the two components of the IRP - routing and inventory holding costs.

We can observe that periodic schedules in the original IRP instance correspond to solutions with some monotonicity property in the PCVRP instance. Hence we can save the approximation factor incurred by transforming PCVRP solutions into periodic schedules if we can compute monotone PCVRP solutions. To do this, we solve a natural linear programming relaxation of the problem strengthened by adding the constraint on monotonicity, and then use randomized threshold rounding for converting fractional solutions to an integral solution. This gives rise to the approximation guarantees that we provide.

We can extend this idea to the capacitated case. When all vehicles have the same capacity, then we can relate the cost of the corresponding capacitated VRP to a lower-bound defined from the shortest length from the root to each client scaled by the ratio of its demands to the capacity. To solve the overall problem, we solve the uncapacitated version of the problem by ignoring the capacity, and break the obtained routes for satisfying the capacity constraints. The key idea is that the costs for connecting the generated sub-routes to the root is bounded by the lower-bound we obtain.

2.3 Summary of Results

For simplicity, in the sequel, we consider the cost of the vehicle route at any period to be modeled by the length of a minimum-cost Steiner tree connecting the depot to the clients that are visited in that period. Our results easily generalize to the case when the vehicle routes are TSP tours over the same set with the appropriate replacement of Steiner tree with TSP in the guarantees below.

A solution x specifies the Steiner trees used by the vehicles at every round, and hence delivers at every visited client, as many units as are needed to satisfy the demand from the current round to the next round when it is visited in the solution. We denote the total holding cost incurred at all clients by the solution x by $h(x)$ and the total routing cost of all vehicles in x by $w(x)$. Let ρ_{PC} denote the best-known approximation factor for the prize-collecting Steiner tree problem (PCST) - the value of ρ_{PC} is currently $(2-\epsilon)$ for some small constant $\epsilon > 0$ [4].

Constant Approximation for Uncapacitated IRP

First we present constant-factor approximation algorithms for uncapacitated IRPs, which are summarized as follows.

1. We design a 2.55-approximation for the minimum-cost nested periodic policy for the uncapacitated IRP.
2. We give an $\alpha_{\mathsf{NE}}\rho_{\mathsf{PC}}$-approximation algorithms for the minimum-cost nested partition and periodic policies for the uncapacitated IRP where α_{NE} is a constant not larger than 2 which is defined in terms of the given frequencies.
3. We design an $\alpha_{\mathsf{NA}}\rho_{\mathsf{PC}}$-approximation algorithm for the minimum-cost nested partition policy for the uncapacitated IRP where α_{NA} is a constant not larger than 2. Note that α_{NA} is a different constant from α_{NE}.
4. We also study general partition policies for uncapacitated IRP which do not need to be nested and provide a $4\rho_{\mathsf{PC}}$-approximation algorithm for the minimum-cost general partition policies. This result is a consequence of the above $\alpha_{\mathsf{NA}}\rho_{\mathsf{PC}}$-approximation for 2-partition policies and a structural relation between 2-partition policies and general partition policies.

In this article, because of the space limitation, we only explain the 2.55-approximation algorithm for the nested periodic policies (Corollary 5) and the $\alpha_{\mathsf{NE}}\rho_{\mathsf{PC}}$-approximation algorithms for the nested partition policies (Theorem 6) in Section 5.

Constant Approximation for IRP with Uniform Capacities

Next we show a general reduction of the capacitated IRP to the corresponding uncapacitated version with a slightly worse guarantee (details are in Section 6).

1. We show that for *any* periodic policy for the IRP, a γ-approximation algorithm for the uncapacitated version can be used to design a $(\gamma+2)$-approximation for the corresponding capacitated version (Theorem 9).
2. For *any* partition policy, we show that a γ-approximation algorithm for the uncapacitated version gives a $(\gamma + 4)$-approximation when each client has a constant demand rate (Theorem 10).

As corollaries, these results yield a counterpart for every result for uncapacitated IRP.

Structural Results – Relations Between Various Types of Policies

Finally we study relations between partition policies and periodic policies. We also relate the optimal value of *any* periodic schedule to that of an optimal 2-periodic schedule for the same instance in the spirit of the result of Roundy [27].

1. We provide an upper bound on the cost of the optimum nested partition policy in terms of the cost of the optimum nested periodic policy for uncapacitated IRP. Formally, we show that $c(x^*_{\mathsf{NE}}) \leq c(x^*_{\mathsf{NA}}) \leq \alpha_{\mathsf{NE}} c(x^*_{\mathsf{NE}})$, where x^*_{NE} and x^*_{NA} respectively denote the optimum nested periodic and partition policies. As a consequence of this result, any ϕ-approximation algorithm for the minimum-cost nested partition policy for uncapacitated IRP is also an $\alpha_{\mathsf{NE}}\phi$-approximation for the corresponding minimum-cost nested periodic policy.
2. We show that for any partition policy x to the uncapacitated IRP, there exists a 2-partition policy y such that $w(y) \leq 2w(x)$ and $h(y) \leq h(x)$. Note that this immediately implies a constant factor approximation algorithm for optimal partition policies with arbitrary frequencies.
3. We do not have an analogous result for arbitrary periodic policies. However, for any k-periodic policy x for $k \geq 3$, we show that there exists a 2-periodic policy y such that $w(y) \leq 2w(x)$ and $h(y) \leq h(x)$.

We do not prove these results in this article due to the space limiatation. We recommend referring to the full version.

2.4 Roadmap

We briefly survey related work in Section 3. Then, in order to illustrate the idea of our reduction from the uncapacitated IRP to the PCVRP, we present an approximation algorithm for the uncapacitated version of 2-periodic or power-of-two policies in Section 4. We present the approximation algorithms for nested policies in uncapacitated IRP in Section 5, and the reductions from the capacitated to the uncapacitated IRP in Section 6. We conclude the paper in Section 7.

3 Related Work

IRP has a vast literature that is addressed in several surveys [20, 11, 28, 29] that focus on different variants of the problem such as those considering stochastic demands, capacitated vehicles or capacities on local client inventories. Solution approaches for these versions in turn can be categorized into three main groups: (i) designing heuristics, (ii) designing policies typically for the infinite-horizon stochastic demand version and showing their (near-)optimality, and most directly related to our work, (iii) designing approximation algorithms for special cases of the problem. We only review the last stream below.

We can model the Vehicle Routing Problem with vehicle capacity constraints by an IRP instance in which we are given a single round and a capacitated vehicle. When the clients demands and the vehicle capacities are uniform, Charikar, Khuller, and Raghavachari [16] gave a 5-approximation. In addition to the VRPs, inventory replenishment problems such as the Joint Replenishment Problem (JRP) can also be seen as a special case of the IRP. In the JRP, there is no metric or clients but several retailers, and each retailer has a local retailer-ordering cost that must be paid whenever it places an order. In addition, there is a warehouse ordering cost that must be paid whenever any retailer places an order. These fixed costs for ordering at any time period can be represented by a simple two level tree: the edge from the root (warehouse) to a dummy node represents the warehouse order cost; edges from the dummy to each client/retailer represents the retailer order cost. A subset of retailers ordering at a round involves paying the cost of the tree induced by them and the root converting this to a case of the IRP on a star network. For the JRP, the first constant approximation ratio was provided by Levi, Round, and Shmoys [25] by giving a 2-approximation algorithm. Later, in [26], they improved the approximation ratio to 1.8. The problem is also studied in the online setting, where the demands are not given in advance, but they arrive online. For this case, a 3-approximation algorithm was provided in [12]. While our work generalizes the JRP to arbitrary metrics and still derives constant-factor approximation algorithms, we focus only on periodic schedules.

4 2-Periodic Policies for Uncapacitated IRP

4.1 Preliminaries

In this section we modify the objective function slightly to remove the contribution of the routing cost in the initial round to simply future calculations.

Note that in round 0 in any partition or periodic policy, all of the client nodes should be visited assuming that there is nonnegative demand in that round (so as to satisfy these demands). We therefore assume without loss of generality that any partition or periodic policy uses an (approximately) optimum Steiner tree in round 0 to visit all of the client nodes.

Let x^*_{PE} denote the optimum 2-periodic policy. Given any weight function $\psi\colon E(G) \to \mathbb{R}$ and a sub-graph H of G, we define $\psi(H)$ to be $\sum_{e\in E(H)} \psi(e)$.

For any policy x, let $h(x)$ and $\bar{w}(x)$ respectively denote the holding and shipping cost incurred in x. Define $\bar{c}(x)$ to be the total cost of the policy, i.e. $\bar{c}(x) = h(x) + \bar{w}(x)$. For any policy x, let $w_0(x)$ be the incurred shipping cost in round 0 of x and define $w(x) = \bar{w}(x) - w_0(x)$ and $c(x) = \bar{c}(x) - w_0(x)$. To simplify the analysis, we define the approximation factor in terms of the refined cost function $c(x)$ (Note that this does not worsen the performance factors).

4.2 Main Idea of the Algorithm

We set up an instance of PCST and solve this instance using an existing approximation algorithm for PCST, e.g. Goemans-Williamson Algorithm (denoted by the GW Algorithm from now on) [22]. Then, using the solution, we construct a policy for the originally given IRP instance.

Here we formally define the PCST instance. Let $L = \lfloor \log T \rfloor$. The instance includes a dummy root node r^*, and $L+1$ copies of G, namely $G_0, \ldots, G_L$. The only difference between G_i and G is the edge weights. Recall that $w(e)$ denotes the weight of an edge e in G. Let

the copy of e in G_i be denoted by e_i, and define the weight of e_i, i.e. $w(e_i)$, as follows:

$$w(e_i) = w(e) \cdot \left\lceil \frac{\lfloor T/2^i \rfloor}{2} \right\rceil. \tag{1}$$

To avoid the risk of confusion, when it is needed, we denote the weight function in the graph G by $w_G(\cdot)$, and the weight function in G_i by $w_i(\cdot)$. For convenience, we sometimes identify $S \subseteq V(G_i)$ with $\{v \in V \mid v_i \in S\}$, and $U \subseteq E(G_i)$ with $\{e \in E(G) \mid e_i \in U\}$.

Let r_i be the copy of the root node r in G_i. Vertices $r_0, \ldots, r_L$ are connected to r^* with edges of weight 0. To finish the definition of the PCST instance, it only remains to define the penalties of nodes. For any node $v \in V(G)$ denote its copy in G_i by v_i. We define the penalties $p(v_i)$ so that $\sum_{j=0}^{i-1} p(v_j)$ will be equal to the total holding cost that node v pays if it is visited with frequency 2^i. More formally, the penalties are defined as follows:

$$p(v_i) = \begin{cases} \mathcal{H}(v,1), & \text{if } i = 0 \\ \mathcal{H}(v,i+1) - \mathcal{H}(v,i), & \text{if } 0 < i \leq L \end{cases} \tag{2}$$

where $\mathcal{H}(v,i)$ will be defined below. Before that, note that with this definition of the penalties, we will have $\sum_{j=0}^{i-1} p(v_j) = \mathcal{H}(v,i)$.

As we mentioned before, we want to set $\mathcal{H}(v,i)$ so that it is equal to the total holding cost that node v pays if it is visited with frequency 2^i. To define this quantity more formally, see that when v is visited every 2^i rounds, then it will be visited in rounds $k \cdot 2^i$ for all integers k such that $k \leq T/2^i$. Hence, when visiting node v in round $k \cdot 2^i$, we should deliver the demand it requires from round $k \cdot 2^i$ to round $\min\{T, (k+1) \cdot 2^i - 1\}$. This will determine the holding cost that v incurs for period $k \cdot 2^i$ to $\min\{T, (k+1) \cdot 2^i - 1\}$. Summing this over all $k \leq T/2^i$ will give $\mathcal{H}(v,i)$, i.e. the total holding cost incurred by v when it is visited every 2^i rounds. Thus, from (2) we get the following proposition:

▶ **Proposition 1.** *For any $i \leq L$ we have $\sum_{j=0}^{i-1} p(v_j) = \mathcal{H}(v,i)$.*

▶ **Proposition 2.** *For any i from 0 to L we have $p(v_i) \geq 0$.*

4.3 2-Periodic Policies

For finding a 2-periodic policy, we need to assign a frequency 2^i to each client node where i can vary from 0 to L. If a node is assigned frequency 2^i, then the policy guarantees to visit it every 2^i rounds and delivers the required demand until the next visit. Note that unlike the partition policy, the node may be reached via the different routes (trees) in every visit. To define a periodic policy completely, we need to define these routes as well.

Let S_i be the set of client nodes that are assigned frequency 2^i. For any round j, the nodes in S_i need to be visited in that round if j is a multiple of 2^i. In other words, if we define $\xi(j)$ to be the largest integer k such that j is a multiple of 2^k, then the nodes in $S_0, \ldots, S_{\xi(j)}$ need to be visited in round j. Consequently, for any round j, the shipping routes are defined by a tree $T_{\xi(j)}$ which visits the nodes in $S_0, \ldots, S_{\xi(j)}$. Below we present an algorithm which provides an approximately optimum 2-periodic policy by outputting the node sets $S_0, \ldots, S_L$, along with the set of trees $T_0, \ldots, T_L$. Again, note that by the definition, the tree T_i visits the nodes in $S_0, \ldots, S_i$.

Algorithm *Periodic Policy*
Input: An IRP instance
Output: A 2-periodic policy defined by subsets $S_0, \ldots, S_L$ of the client node set and trees $T_0, \ldots, T_L$

1. **for** $i = 0$ **to** L
2. **do** $S_i \leftarrow \emptyset$
3. Construct the PCST instance.
4. Solve the PCST instance.
5. For all i, let Q_i be the set of client nodes in G_i which, using the tree U_i, got connected to r^* in the solution.
6. **for** $i = 0$ **to** L
7. **do** $S_i \leftarrow Q_i \setminus \cup_{j=0}^{i-1} Q_j$
8. Let H_i be the subgraph of G with the edge set $E(U_0) \cup \ldots \cup E(U_i)$.
9. Let T_i be an arbitrary spanning tree in H_i.
10. Output $S_0, \ldots, S_L$ and $T_0, \ldots, T_L$.

▶ **Lemma 1.** *In any periodic policy, and for any integer i such that $i \leq L$, the tree T_i is used in exactly $\left\lceil \frac{\lfloor T/2^i \rfloor}{2} \right\rceil$ number of the rounds (recall that round 0 is excluded).*

Proof. Since for each round j we use the tree $T_{\xi(j)}$, then the number of times that the tree T_i is used is equal to the number of integers p such that $1 \leq p \leq T$ and $p/2^i$ is an odd integer. It is easy to verify that there are exactly $\left\lceil \frac{\lfloor T/2^i \rfloor}{2} \right\rceil$ values which can be assigned to p. ◀

▶ **Theorem 2.** *If Step 4 of Algorithm Periodic Policy uses a ρ_{ST}-approximation algorithm for solving the PCST instance, then the algorithm achieves approximation factor $2\rho_{\mathsf{PC}}$. If the GW algorithm is used instead of the ρ_{ST}-approximation algorithm, then Algorithm Periodic Policy finds a periodic policy x_{PE} such that $c(x_{\mathsf{PE}}) \leq 2h(x^*_{\mathsf{PE}}) + 4w(x^*_{\mathsf{PE}})$.*

Proof. Let y^* denote the optimum solution for the constructed PCST instance. Let $p(y^*)$, $w(y^*)$, and $c(y^*)$ respectively denote the penalty cost, the tree cost, and the total cost of y^*. We hence have $c(y^*) = p(y^*) + w(y^*)$.

The proof consists of three steps. In the first step, we prove that $p(y^*) \leq h(x^*_{\mathsf{PE}})$ and $w(y^*) \leq w(x^*_{\mathsf{PE}})$. Then, in the second step, we observe that Step 4 of Algorithm *Periodic Policy* finds a solution $\hat{y}$ of cost at most $\rho_{\mathsf{ST}} c(y^*)$ when it uses a ρ_{ST}-approximation algorithm for PCST. In fact, this needs no proof since it follows from the definition of ρ_{ST}-approximation. When Step 4 uses the GW algorithm, then it finds a solution $\hat{y}$ of cost at most $p(y^*) + 2w(y^*)$, which was proven in [22]. Finally, in the third step, we show that Algorithm *Periodic Policy* converts $\hat{y}$ into a periodic policy x_{PE} such that $c(x_{\mathsf{PE}}) \leq 2c(\hat{y})$.

To do the first step, given x^*_{PE}, we construct a solution y for the PCST instance such that $p(y) = h(x^*_{\mathsf{PE}})$ and $w(y) = w(x^*_{\mathsf{PE}})$. Let $S^*_0, \ldots, S^*_L$ be the subsets of the client node set in the periodic policy x^*_{PE}, and let $T^*_0, \ldots, T^*_L$ respectively be their associated trees. Then, construct y as follows: for each copy G_i in the PCST instance, visit the nodes in $\cup_{j=0}^{i} S^*_j$ using the tree T^*_i, and pay the penalty for the nodes in $V(G_i) \setminus (\cup_{j=0}^{i} S^*_j)$. Observe that every node $v \in S^*_i$ is visited in all of the copies except $G_0, \ldots, G_{i-1}$. We have $\sum_{j=0}^{i-1} p(v_j) = \mathcal{H}(v, i)$ by Proposition 1. These mean that the overall holding cost paid for v in x^*_{PE} is equal to the overall penalty paid for (the copies of) v in y. Therefore $p(y) = h(x^*_{\mathsf{PE}})$ holds.

It is easy to see that $w(y) = w(x^*_{\mathsf{PE}})$ holds as well. Just observe that the number of times that the tree T^*_i is used in x^*_{PE} is equal to $\left\lceil \frac{\lfloor T/2^i \rfloor}{2} \right\rceil$ by Lemma 1, and hence the total tree

cost paid for using T_i^* in x^*_{PE} is $w_G(T_i^*) \cdot \left\lceil \frac{\lfloor T/2^i \rfloor}{2} \right\rceil$, which is exactly equal to $w_i(T_i)$, i.e. the tree cost incurred in y for copy G_i. Summing over all i implies that $w(y) = w(x^*_{\mathsf{PE}})$ holds.

In the last step of the proof, we show that $c(x_{\mathsf{PE}}) \leq 2c(\hat{y})$ by showing that $h(x_{\mathsf{PE}}) \leq p(\hat{y})$ and $w(x_{\mathsf{PE}}) \leq 2w(\hat{y})$. To prove $h(x_{\mathsf{PE}}) \leq p(\hat{y})$, fix a node v and let i be the smallest integer such that v_i is connected to r^* in $\hat{y}$. Hence the overall penalty paid for (the copies of) v would be at least $\mathcal{H}(v,i)$ by Lemma 1. On the other hand, by the choice of S_i in Algorithm *Periodic Policy*, we have $v \in S_i$. This guarantees that the overall holding cost paid for v in x_{PE} is exactly $\mathcal{H}(v,i)$. By the two latter facts, the overall holding cost paid for v in x_{PE} is at most the overall penalty paid for v in $\hat{y}$. By summing over all v, we get $h(x_{\mathsf{PE}}) \leq p(\hat{y})$.

It remains to show that $w(x_{\mathsf{PE}}) \leq 2w(\hat{y})$. By the choice of $T_0, \ldots, T_L$ in Algorithm *Periodic Policy*,

$$w(x_{\mathsf{PE}}) \leq \sum_{i=0}^{L} \sum_{j=i}^{L} w_j(U_i). \tag{3}$$

Now, if for any fixed i, we show that $\sum_{j=i+1}^{L} w_j(U_i) \leq w_i(U_i)$, then by (3) we have

$$w(x_{\mathsf{PE}}) \leq \sum_{i=0}^{L} \sum_{j=i}^{L} w_j(U_i) \leq \sum_{i=0}^{L} 2w_i(U_i) = 2w(\hat{y}) \tag{4}$$

where the equality in (4) is due to the fact that $\sum_{i=0}^{L} w_i(U_i) = w(\hat{y})$. Thus it only remains to show that $\sum_{j=i+1}^{L} w_j(U_i) \leq w_i(U_i)$. Equivalently, by the definition of $w_i(\cdot)$, we have to show that

$$\sum_{j=i+1}^{L} \left\lceil \frac{\lfloor T/2^j \rfloor}{2} \right\rceil \leq \left\lceil \frac{\lfloor T/2^i \rfloor}{2} \right\rceil.$$

This inequality can be proven by elementary calculations. ◀

5 Nested Policies for Uncapacitated IRP

In this section, we generalize the context of

Section 4 from power-of-two to arbitrary nested policies. We also refine the method used in Section 4 to convert our problem to a *monotone* version of a prize-collecting VRP. In particular, we present two approximation results for nested policies, one of which is for nested periodic policies, and the other of which is for nested partition policies.

Let $q_0, q_1, \ldots, q_k$ be integers such that $q_0 = 1$ and $q_j \geq 2$ for $j = 1, 2, \ldots, k$. In a nested policy, available frequencies are $f_0, f_1, \ldots, f_{k+1}$ that are defined as $f_i = \prod_{j=0}^{i} q_j$ for $i = 0, 1, \ldots, k$, and $f_{k+1} = +\infty$.

5.1 2.55-Approximation Algorithm for Nested Periodic Policies

We here present an algorithm for nested periodic policies. Our algorithm again reduces the problem to PCST as Algorithm *Periodic Policy* in Section 4 did. In our reduction, the graph and penalties are same as before (we replace L by k); The graph is the union of $k+1$ copies $G_0, G_1, \ldots, G_k$ of G and a new node r^* connected to the copies of r by edges of weight 0; The penalty $p(v_i)$ of the i-th copy of a client node v is defined by (2) where $\mathcal{H}(v,i)$ is the total holding cost that node v pays when it is assigned frequency f_i. We define the weight

$w(e_i)$ of the i-th copy of an edge e as the one we need to pay when we use it in a tree of frequency f_i, as follows.

$$w_i(e_i) = w_G(e)\left(\left\lfloor \frac{T}{f_i} \right\rfloor - \left\lfloor \frac{T}{f_{i+1}} \right\rfloor\right). \tag{5}$$

We say that a solution for the PCST instance is *monotone* when for any client node v and for any i and j such that $1 \leq i < j \leq k$, v_j is connected to r^* if v_i is connected to r^*. In our algorithm for nested periodic policies, we have to approximate a minimum cost monotone solution for the PCST instance. We here assume that there exists a ρ-approximation algorithm for this problem. At the end of this subsection, we mention that there exists an algorithm with $\rho < 2.55$. The construction of a nested periodic policy from a monotone solution for PCST is almost same as *Periodic Policy*; A client node v is assigned frequency f_i when i is the minimum index such that v_i is connected to r^* in the monotone solution for PCST. We use the tree of the solution in G_i to visit client nodes of frequency at most f_i in round t such that f_i is the maximum frequency that divides t.

In the next theorem, we show that the modified *Periodic Policy* is a ρ-approximation algorithm. We omit its proof due to the space limitation.

▶ **Theorem 3.** *Suppose that there exists a ρ-approximation algorithm for finding a minimum cost monotone solution for the PCST instance defined above. Then the problem of fining a minimum cost nested periodic policy for uncapacitated IRP admits a ρ-approximation algorithm.*

Let us discuss algorithms for approximating minimum cost monotone solutions for PCST. The algorithm due to [4] achieves $\rho_{\mathsf{PC}} = 2 - \epsilon$ for some constant ϵ currently, and the GW algorithm [22] achieves approximation factor 2 for PCST. We do not know if these algorithms can be modified for approximating monotone solutions. What we can do here is to modify the algorithm due to an unpublished work of Goemans (refer to [23, 32]). This algorithm achieves approximation factor $1/(1 - e^{-1/2}) < 2.55$ as follows: Consider an LP relaxation of PCST which has a variable $x(e)$ for representing what fraction of an edge e is chosen in a solution, and a variable $y(v)$ for representing what fraction of a terminal v is covered by the solution; The algorithm solves the LP relaxation to obtain an optimal solution (x^*, y^*); It also chooses a threshold α uniformly at random from $[e^{-1/2}, 1]$, and let $\hat{S} = \{v \mid y^*(v) \geq \alpha\}$; The algorithm outputs a Steiner tree that connects $\hat{S}$ to the root. The LP used there still gives a lower-bound on the optimal value of our problem even if we add a new constraint

$$y(v_0) \leq y(v_1) \leq \cdots \leq y(v_k)$$

for each $v \in V$, and by this new constraint, the Steiner tree output by the algorithm is monotone. It is not difficult to verify that this Steiner tree achieves the same approximation factor as before, and we therefore have the following theorem.

▶ **Theorem 4.** *The problem of finding a minimum cost monotone PCST admits an approximation factor within $1/(1 - e^{-1/2}) < 2.55$.*

Theorems 3 and 4 gives the next corollary.

▶ **Corollary 5.** *The problem of finding a minimum cost nested periodic policy for uncapacitated IRP can be approximated within a factor of $1/(1 - e^{-1/2}) < 2.55$.*

5.2 $\alpha_{\mathsf{NE}}\rho_{\mathsf{PC}}$-Approximation Algorithm for Nested Partition Policies

For approximating nested partition policies, we use the same reduction to PCST as for nested periodic policies. When we solve the constructed instance of PCST, we do not have to restrict solutions to monotone solutions here, and hence we can use a ρ_{PC}-approximation algorithm.

Let y be a solution for PCST computed by the ρ_{PC}-approximation algorithm. Let T_i be the subtree of y in G_i. While T_i is used $\lfloor T/f_i \rfloor - \lfloor T/f_{i+1} \rfloor$ times for visiting $S_0 \cup S_1 \cup \cdots \cup S_i$ in nested periodic policies, our nested partition policy uses it $\lfloor T/f_i \rfloor$ times for visiting only S_i because a client node in S_i has to be reached via the same tree at every visit.

Let

$$\alpha_{\mathsf{NE}} = \max_{1 \le i \le k} \left(1 + \frac{1}{q_i - 1}\right). \tag{6}$$

We always have $\alpha_{\mathsf{NE}} \le 2$ since $q_i \ge 2$ for $i = 1, 2, \ldots, k$.

▶ **Theorem 6.** *Suppose that there exists a ρ_{PC}-approximation algorithm for PCST. Then we can compute a nested partition policy x_{NA} such that $h(x_{\mathsf{NA}}) + w(x_{\mathsf{NA}}) \le \rho_{\mathsf{PC}} h(x^*_{\mathsf{NE}}) + \alpha_{\mathsf{NE}}\rho_{\mathsf{PC}} w(x^*_{\mathsf{NE}})$. In particular, the problem of finding a minimum cost nested partition policy for uncapacitated IRP admits an $\alpha_{\mathsf{NE}}\rho_{\mathsf{PC}}$-approximation algorithm.*

Proof. Let y be a ρ_{PC}-approximate solution for the PCST instance. Let x_{NA} be the nested partition policy computed from y by our algorithm. We compare x_{NA} with an optimal nested periodic policy x^*_{NE}. This is enough because the minimum cost of nested periodic policies is at most that of nested partition policies.

In the proof of Theorem 3, we have already proven that there exists a monotone solution $\hat{y}$ for PCST such that $w(\hat{y}) = w(x^*_{\mathsf{NE}})$ and $p(\hat{y}) \le h(x^*_{\mathsf{NE}})$. Since the minimum cost of any solutions for PCST is at most $w(\hat{y}) + p(\hat{y})$, we have $p(y) + w(y) \le \rho'(h(x^*_{\mathsf{NE}}) + w(x^*_{\mathsf{NE}}))$. We can also verify that $h(x_{\mathsf{NA}}) \le p(y)$ holds as in the proof of Theorem 3. For proving $w(x_{\mathsf{NA}}) \le \alpha_{\mathsf{NE}} w(y)$, it suffices to show

$$\left\lfloor \frac{T}{f_i} \right\rfloor \le \alpha_{\mathsf{NE}} \left(\left\lfloor \frac{T}{f_i} \right\rfloor - \left\lfloor \frac{T}{f_{i+1}} \right\rfloor \right). \tag{7}$$

Notice that

$$\frac{1}{q_{i+1}-1} \cdot \left\lfloor \frac{T}{f_i} \right\rfloor = \left(\frac{q_{i+1}}{q_{i+1}-1}\right) \cdot \frac{1}{q_{i+1}} \cdot \left\lfloor \frac{T}{f_i} \right\rfloor \ge \frac{q_{i+1}}{q_{i+1}-1} \cdot \left\lfloor \frac{T}{q_{i+1} f_i} \right\rfloor = \frac{q_{i+1}}{q_{i+1}-1} \cdot \left\lfloor \frac{T}{f_{i+1}} \right\rfloor.$$

This inequality is equivalent to

$$\left\lfloor \frac{T}{f_{i+1}} \right\rfloor \le \frac{1}{q_{i+1}-1} \cdot \left(\left\lfloor \frac{T}{f_i} \right\rfloor - \left\lfloor \frac{T}{f_{i+1}} \right\rfloor \right),$$

and the definition of α_{NE} gives

$$\frac{1}{q_{i+1}-1} \cdot \left(\left\lfloor \frac{T}{f_i} \right\rfloor - \left\lfloor \frac{T}{f_{i+1}} \right\rfloor \right) \le (\alpha_{\mathsf{NE}} - 1) \left(\left\lfloor \frac{T}{f_i} \right\rfloor - \left\lfloor \frac{T}{f_{i+1}} \right\rfloor \right).$$

Combining these inequalities gives the required one. ◀

6 Reducing Capacitated IRP to its Uncapacitated Version

In this section, we consider capacitated IRP. Because of the capacity constraints, we may need more than one tree for visiting nodes in a single round. We assume that these trees used in the same round can share an edge while we need to pay its weight multiple times. When a client node is connected to the root by more than one tree, we have to specify which tree takes care of the demand of this node. In other words, a schedule for a single round consists of a set of trees and an allocation of each client node to one of these trees. The capacity constraints require that the total demand of nodes assigned to a tree does not exceed a given capacity C.

Our main result in this section is a reduction of the capacitated problem to the corresponding uncapacitated version with a slight worsening in the performance ratio. Recall that the vehicle routing cost component of the IRP we model is the minimum-cost Steiner tree rather than the tour. The reduction below applies to other IRPs where the routing is via a tour on the client nodes, but with slightly different factors.

First we present a lower-bound on the tree costs of feasible solutions.

▶ **Lemma 7.** *Let x_C denote any capacitated IRP solution with vehicle capacities C, and $w(r,v)$ denote the weight of the direct edge between r and v in the given metric. Then $w(x_C) \geq \sum_v w(r,v) \sum_{t=0}^{T} d_v^t/C$ (recall that d_v^t is the demand of client v in round t).*

Proof. Consider how every unit of demand to any client is delivered from r in x_C. We "charge" the tree path in the solution from the root to the client scaled by $1/C$ to that unit of demand. Since there are at most C units of demands in any tree, no edge of x gets charged more than once and the paths charged between client i and r have weights at least $w(r,i)$ by the metric property. ◀

We also need the following lemma on partitioning a tree.

▶ **Lemma 8.** *Let U be a rooted tree, and let S denote a set of nodes spanned by U. Suppose that each $v \in S$ has an integer D_v such that $0 \leq D_v \leq C$. Then U can be partitioned into edge-disjoint subtrees $U_1, U_2, \ldots, U_\ell$, and each $v \in S$ can be allocated to one of the subtrees so that*

- **(i)** *$v \in S$ is allocated to the subtree that spans it,*
- **(ii)** *$\sum_{v \in S(U_j)} D_v \leq C$ for each $j = 1, 2, \ldots, \ell$ where $S(U_j)$ denotes the set of nodes in S allocated to U_j,*
- **(iii)** *$\sum_{v \in S(U_j)} D_v \geq C/2$ holds if U_j does not span the root.*

Proof. We prove the lemma by the induction on $|S|$. For $v \in S$, let S_v denote the set of descendants of v in S, and U_v denote the subtree of U which is induced by v and its descendents. Let v^* be a node farthest from the root such that $D_{v^*} + \sum_{v \in S_{v^*}} D_v > C$. If there exists no such v^* (including the case of $|S| = 0$), then we are done.

Suppose not. Then $D_{v^*} > C/2$ or $\sum_{v \in S_{v^*}} D_v > C/2$ holds. Notice that $\sum_{v \in S_{v^*}} D_v \leq C$ holds by the definition of v^*, and $D_{v^*} \leq C$ by the assumption. When the former condition holds, we define the subtree that consists of only v^*, and allocate v^* to this subtree. We then remove v^* from S and apply the induction. When the latter condition holds, we let U_{v^*} be one of the subtrees, and allocate nodes in S_{v^*} to U_{v^*}. We then remove the edges in U_{v^*} from U, and apply the induction. ◀

▶ **Theorem 9.** *Given a γ-approximation for the uncapacitated version of the minimum cost periodic IRP, there is a $(\gamma + 2)$-approximation for the corresponding capacitated version where every tree supports demand at most the given capacity.*

We emphasize that the above reduction applies to all periodic policies, in particular to nested periodic policies. The proof proceeds using a natural combination of the uncapacitated solution along with shortcuts to replenish the supply whenever the vehicle routing solution runs out due to its capacity constraint [1, 24].

Proof. First note that an optimal solution to the uncapacitated counterpart of the given capacitated IRP provides a lower bound on the optimal value of the original capacitated version as well. We first apply the given γ-approximation ignoring the capacities to get an uncapacitated periodic solution to the IRP. Note that this solution defines, without loss of generality, a single tree U that connects the root with all the clients that must be visited in this round.

Let v be a client visited by U in the uncapacitated periodic solution. We assume that a vehicle does not have to deliver more than C units to v in this single visit[1]. We can assume without loss of generality that the uncapacitated solution has this property because we can set $h^v_{st} = +\infty$ when we apply the γ-approximation algorithm if v demands more than C units in rounds from s to t. Since any feasible solutions for the capacitated instance also has the property, this transformation of h^v_{st} makes no effect on the above claim that the uncapacitated solution provides a lower-bound on the optimal value. We define D_v as the units of demands delivered to v by U in the uncapacitated solution. The above assumption implies that $D_v \leq C$.

To complete the algorithm, we need to break every tree U in the uncapacitated solution for the periodic IRP into trees of capacity at most C each. To do this, we employ Lemma 8. Then U is broken into subtrees $U_1, U_2, \ldots, U_\ell$, and each client is allocated to one of the subtrees. For a subtree U_j, let $S(U_j)$ be the set of clients allocated to U_j. If U_j does not span the root, we add the cheapest edge ru_j from a node $u_j \in S(U_j)$ as the "connector" edge to the root to build our capacitated trees of capacity at most C.

Since the subtrees are edge-disjoint, it suffices to show that the weights of the connector edges for all the subtrees can be bounded by twice the lower-bound given in Lemma 7 to get the final guarantee of $\gamma + 2$. For this, observe that we have

$$\sum_{v \in S(U_j)} D_v w(r,v) \geq w(r,u_j) \sum_{v \in S(U_j)} D_v \geq w(r,u_j) \cdot \frac{C}{2},$$

where the last inequality is due to the condition (iii) in Lemma 8. The above inequality simplifies to $w(r,u_j) \leq 2 \sum_{v \in S(U_j)} D_v w(r,v)/C$. A unit of demand is not assigned to more than one subtree simultaneously. This means that the total weight of connector edges is at most twice the lower-bound in Lemma 7. ◀

In order to approximate partition policies, we have to assume that each client v has a constant demand rate d_v per round for a technical reason. Note that several papers [8, 9] assume a constant demand rate per round for the IRP under which all solutions of the same frequency route the same amount of demand in every visit to the same node.

▶ **Theorem 10.** *Given a γ-approximation for the uncapacitated version of the minimum cost partition IRP, there is a $(\gamma + 4)$-approximation for the corresponding capacitated version with constant demand rates where every tree supports demand at most the given capacity.*

[1] If we are allowed to split the delivery of the demands to a single node by multiple visits of different capacitated vehicles in the same round, our method can be modified to handle this case with an even better guarantee; we omit discussion of this easier case.

Proof. In Theorem 9, the partitions of trees are possibly different even if the trees visit the same set of clients because the demands of a client can change in different rounds that we employ the tree. Such different partitions are disallowed as solutions to partition policies. In the current setting, this does not happen because a client v of frequency f_i always demands $f_i d_v$ units unless it is at a round in $[T - f_i + 1, T]$. Hence we take the following approach.

When we partition a tree U of frequency f_i, we apply Lemma 8 with $D_v = f_i d_v$ for each client v even if U is used at a round in $[T - f_i + 1, T]$. This way, we always have the same partition for trees of the same frequency in different rounds. Call a tree U the *last tree* if it is used at a round in $[T - f_i + 1, T]$. It can be proven as before that the total weights of connector edges used for augmenting subtrees constructed from trees that are not last trees can be bounded by twice the lower-bound in Lemma 7. For bounding the weights of connector edges used for augmenting subtrees constructed from the last trees, we re-charge units demanded in all rounds. This are at least $D_v = f_i d_v$ units of demands for each client v because $f_i \leq T$. Since a client is not contained by more than one last tree, we do not overuse the demands more than once. Hence the weights of the connector edges for the last trees is also at most twice the lower-bound in Lemma 7. In total, four times the lower-bound is enough for paying the weights of connector edges. ◀

7 Conclusion

We presented constant factor approximation algorithms for finding minimum cost periodic schedules in IRP. A natural question is whether efficient algorithms exist for finding non-periodic schedules. More formally, the problem with non-periodic schedules is defined as follows. For every period in the horizon, we can design a separate tree or tour routes, and the demand for any client at any time is delivered in the last visit before that time to the client in the set of routes. This is an interesting extension of the classic Steiner tree problem and TSP to the round model. It is not difficult to design an $O(\log |V|)$-approximation algorithm for this problem by reducing to the instances with tree metrics using the metric embedding technique [5]. It is an attractive open question to ask if this problem admits a constant factor approximation algorithm. We hope that our ideas presented in the current paper are useful for obtaining an answer to this question.

Acknowledgements. The first author is supported in part by Japan Society for the Promotion of Science (JSPS), Grants-in-Aid for Young Scientists (B) 25730008. The second author is supported in part by grant FA9550-12-1-0411 from the U.S. Air Force Office of Scientific Research (AFOSR) and the Defense Advanced Research Projects Agency (DARPA). The third author is supported in part by NSF grant CCF-1218382.

References

1 K. Altinkemer, B. Gavish. Heuristics with constant error guarantees for the design of tree networks. Management Science 34 (1988) 331–341.

2 S. Anily, A. Federgruen. One warehouse multiple retailer systems with vehicle routing costs. Management Science 36 (1990) 92–114.

3 S. Anily, A. Federgruen. Two-echelon distribution systems with vehicle routing costs and central inventories. Oper. Res. 41 (1993) 37–47.

4 A. Archer, M.H. Bateni, M.T. Hajiaghayi, H. Karloff. Improved approximation algorithms for prize-collecting Steiner tree and TSP, SIAM J. Comput. 40 (2011) 309–332.

5 J. Fakcharoenphol, S. Rao, K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. J. Comput. Syst. Sci. 69 (2004) 485–497.

6 E. Balas. The prize collecting traveling salesman problem. Networks 19 (1989) 621–636.

7 W.J. Bell, L.M. Dalberto, M.L. Fisher, A.J. Greenfield, R. Jaikumar, P. Kedia, R.G. Mack, P.J. Prutzman. Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. Interfaces 13 (1983) 4–23.

8 L. Bertazzi, M.G. Speranza. Continuous and discrete shipping strategies for the single link problem. Transportation Science 36 (2002) 314–325.

9 L. Bertazzi. Analysis of direct shipping policies in an inventory-routing problem with discrete shipping times. Management Science 54 (2008) 748–762.

10 D. Bienstock, M.X. Goemans, D. Simchi-Levi, D.P. Williamson. A note on the prize collecting traveling salesman problem. Math. Prog. 59 (1993) 413–420.

11 J. Bramel and D. Simchi-Levi. A location based heuristic for general routing problems. Oper. Res. 43 (1997) 649–660.

12 N. Buchbinder, T. Kimbrel, R. Levi, K. Makarychev, M. Sviridenko. Online make-to-order joint replenishment model: primal dual competitive algorithms. SODA 2008, 952–961.

13 L.D. Burns, R.W. Hall, D.E. Blumenfeld, C.F. Daganzo. Distribution strategies that minimize transportation and inventory costs. Oper. Res. 33 (1985) 469–490.

14 A. Campbell, M. Savelsbergh. A decomposition approach for the inventory routing problem. Transportation Science 38 (2004) 488–502.

15 L.M.A. Chan, A. Federgruen, D. Simchi-Levi. Probabilistic analyses and practical algorithms for inventory-routing models. Oper. Res. 46 (1998) 96–106.

16 M. Charikar, S. Khuller, B. Raghavachari. Algorithms for capacitated vehicle routing. SIAM J. Comput. 31 (2001) 665–682.

17 T. Chien, A. Balakrishnan, R. Wong. An integrated inventory allocation and vehicle routing problem. Transportation Science 23 (1989) 67–76.

18 L.C. Coelho, J.-F. Cordeau, G. Laporte, Thirty years of inventory-routing, Transportation Science 48 (2014) 1–9.

19 M. Dror, M. Ball, B. Golden. Computational comparison of algorithms for the inventory routing problem. Annals of Operations Research 4 (1985) 3–23.

20 A. Federgruen, D. Simchi-Levi. Analytical analysis of vehicle routing and inventory management problems. M. O. Ball, T. L. Magnanti, C. L. Monma, G. L. Nemhauser, eds. Network Routing. Handbooks in OR and MS, Vol. 8. North-Holland, Amsterdam, The Netherlands, 297–373 (1995).

21 A. Federgruen, P. Zipkin. A combined vehicle routing and inventory allocation problem. Oper. Res. 32 (1984) 1019–1037.

22 M.X. Goemans, D.P. Williamson, A general approximation technique for constrained forest problems, SIAM J. Comput. 24 (1995) 296–317.

23 M.T. Hajiaghayi, K. Jain. The prize-collecting generalized steiner tree problem via a new approach of primal-dual schema. SODA 2006, 631–640.

24 R. Hassin, R. Ravi, F. S. Salman. Approximation algorithms for a capacitated network design problem, Algorithmica 38 (2004) 417–431.

25 R. Levi, R.O. Round, D.B. Shmoys. Primal-dual algorithms for deterministic inventory problems. Mathematics of Operations Research 31 (2006) 267–284.

26 R. Levi, R. Roundy, D. Shmoys, M. Sviridenko. First constant approximation algorithm for the one-warehouse multi-retailer problem. Management Science 54 (2008) 763–776.

27 R. Roundy. 98%-effective integer-ratio lot-sizing for one-warehouse, multi-retailer systems. Management Science 31 (1985) 1416–1430.

28 A.M. Sarmiento, R. Nagi. A review of integrated analysis of production distribution systems. IIE Transactions 31 (1999) 1061–1074.

29 P. Toth, D. Vigo. The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, PA (2002).

30 Vender Managed Inventory. http://www.vendormanagedinventory.com/

31 S. Viswanathan, K. Mathur, Integrating routing and inventory decisions in one warehouse multi-retailer multi-product distribution systems. Management Science 43 (1997) 294–312.

32 D.P. Williamson, D.B. Shmoys. The Design of Approximation Algorithms. Cambridge University Press (2011).

Complexity and Approximation of the Continuous Network Design Problem

Martin Gairing[1], Tobias Harks[2], and Max Klimm[3]

1 **Department of Computer Science, University of Liverpool, UK**
m.gairing@liverpool.ac.uk
2 **Department of Quantitative Economics, Maastricht University, The Netherlands**
t.harks@maastrichtuniversity.nl
3 **Institut für Mathematik, Technische Universität Berlin, Germany**
klimm@math.tu-berlin.de

Abstract

We revisit a classical problem in transportation, known as the *continuous (bilevel) network design problem*, CNDP for short. Given a graph for which the latency of each edge depends on the ratio of the edge flow and the capacity installed, the goal is to find an optimal investment in edge capacities so as to minimize the sum of the routing cost of the induced Wardrop equilibrium and the investment cost for installing the capacity. While this problem is considered as challenging in the literature, its complexity status was still unknown. We close this gap showing that CNDP is strongly NP-complete and APX-hard, both on directed and undirected networks and even for instances with affine latencies. As for the approximation of the problem, we first provide a detailed analysis for a heuristic studied by Marcotte for the special case of *monomial* latency functions (Math. Program., Vol. 34, 1986). We derive a closed form expression of its approximation guarantee for *arbitrary* sets of latency functions. We then propose a different approximation algorithm and show that it has the same approximation guarantee. However, we show that using the better of the two approximation algorithms results in a strictly improved approximation guarantee for which we derive a closed form expression. For affine latencies, e. g., this algorithm achieves a $49/41 \approx 1.195$-approximation which improves on the 5/4 that has been shown before by Marcotte. We finally discuss the case of hard budget constraints on the capacity investment.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Bilevel optimization, Optimization under equilibrium constraints, Network design, Wardrop equilibrium, Computational complexity, Approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.226

1 Introduction

The continuous network design problem (CNDP) introduced by Dafermos [7], Dantzig et al. [9], and Abdulaal et al. [1] is one of the most classical network design problems in transport. In a nutshell, given a graph in which the latency of each edge depends on the ratio of the edge flow and the capacity installed at that edge, the goal is to find an optimal investment in edge capacities so as to minimize the sum of the routing cost of the induced Wardrop equilibrium and the investment cost for installing the capacity. The investment cost is assumed to be linear in chosen capacity and comprises all monetary costs for building the streets with the given capacity (spread over the expected lifespan of the street) and the accumulated maintenance cost during that time. By a scaling capacity costs accordingly, an arbitrary

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 226–241

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

linear combinations of routing cost and investment cost can be minimized. Continuous network design is a fundamental problem in traffic and telecommunication networks when new networks have to be designed from scratch, e. g., after introducing new technology or after opening up new areas.

From a mathematical perspective, CNDP is a bilevel optimization problem (cf. [5, 16] for an overview), where in the upper level the edge capacities are determined and, given these capacities, in the lower level the flow will settle into a Wardrop equilibrium in which, for each commodity, only shortest paths are used. Clearly, the lower level reaction depends on the upper level decision because altering the capacity investment on a subset of edges may result in revised route choices by users.

CNDP has been intensively studied since the late sixties (cf. [7, 17]) and several heuristic approaches have been proposed since then; see Yang et al. [27] for a comprehensive survey. Most of the proposed heuristics are numerical in nature and involve iterative computations of relaxations of the problem (for instance the iterative optimization and assignment algorithm as described in [19] and augmented Lagrangian methods or linearizations of the objective in the leader and follower problem). An exception is the work of Marcotte [18] who considered several algorithms based on solutions of associated convex optimization problems which can be solved in polynomial time [11]. He derives worst-case bounds for his heuristics and, in particular, for affine latency functions he devises an approximation algorithm with an approximation factor of 5/4. For general monomial latency functions plus a constant (including the latency functions used by the Bureau of Public Roads [25]) he obtains a polynomial time 2-approximation. Variants of CNDP were also considered in the networking literature, see [4, 12, 13, 14]. These works, however, consider the case where a budget capacity must be distributed among a set of edges to improve the resulting equilibrium. Most results, however, only work for simplified network topologies (e. g., parallel links) or special latency functions (e. g., $M/M/1$ latency functions).

1.1 Our Results and Used Techniques.

Despite more than forty years of research, to the best of our knowledge, the computational complexity status of CNDP is still unknown. We close this gap as we show that CNDP is strongly NP-complete and APX-hard, both on directed and undirected networks and even for instances with affine latencies of the form $S_e(v_e/z_e) = \alpha_e + \beta_e \cdot (v_e/z_e)$, where v_e is the flow and z_e the capacity of edge e and $\alpha_e, \beta_e \geq 0$. For the proof of the NP-hardness, we reduce from 3-SAT. The reduction has the property that in case that the underlying instance of 3-SAT has a solution the cost of an optimal solution is equal to the minimal cost of a relaxation of the problem, in which the equilibrium conditions are relaxed. The key challenge of the hardness proof is to obtain a lower bound on the optimal solution when the underlying 3-SAT instance has no solution. To this end, we relax the equilibrium conditions only partially which enables us to bound the cost of an optimal solution from below by solving an associated constrained quadratic optimization problem. With a more involved construction and a more detailed analysis, we can even prove APX-hardness of the problem. Here, we reduce from a symmetric variant of MAX-3-SAT, in which all literals occur exactly twice. While all our hardness proofs rely on instances with an arbitrary number of commodities and respective sinks, we show that for instances in which all commodities share a common sink, CNDP can be solved to optimality in polynomial time.

In light of the hardness of CNDP, we focus on approximation algorithms. We first consider a polynomial time algorithm proposed by Marcotte [18]. This algorithm, which we call BringToEquilibrium, first computes a relaxation of CNDP by removing the

Table 1 Approximation guarantees of the algorithms BringToEquilibrium, ScaleUniformly, and the best of the two for convex latency functions, concave latency functions and sets of polynomials with non-negative coefficients depending on the maximal degree Δ. The approximation guarantees stated for convex latency functions even hold for sets of semi-convex latency functions as in Assumption 2.1. For BringToEquilibrium, the approximation guarantees marked with (*) have been obtained before in [18].

Functions	Approximation guarantees: BringToEquilibrium / ScaleUniformly	Better of the two
concave	$5/4 = 1.25$	$49/41 \approx 1.195$
convex	2	$9/5 = 1.8$
polynomials Δ		
0	1	1
$1/4$	$3381/3125 \approx 1.082$	≈ 1.064
$1/3$	$283/256 \approx 1.105$	≈ 1.083
$1/2$	$31/27 \approx 1.148$	$1849/1657 \approx 1.116$
1	$5/4 = 1.25$ *	$49/41 \approx 1.195$
2	$1 + \frac{2}{9}\sqrt{3} \approx 1.385^\star$	$\frac{311}{479} + \frac{180}{479}\sqrt{3} \approx 1.300$
3	$1 + \frac{3}{16}\sqrt[3]{4^2} \approx 1.472^\star$	≈ 1.369
4	$1 + \frac{4}{25}\sqrt[4]{5^3} \approx 1.535^\star$	≈ 1.418
∞	2 *	$9/5 = 1.8$

equilibrium conditions. Then, it reduces the edge capacities individually such that the flow computed in the relaxation becomes a Wardrop equilibrium. We give a novel closed form expression of the performance of this algorithm with respect to the set $\mathcal{S}$ of allowed latency functions. Specifically, we show that this algorithm is a $(1 + \mu(\mathcal{S}))$-approximation, where $\mu(\mathcal{S}) = \sup_{S\in\mathcal{S},x\geq 0,\gamma\in[0,1]} \gamma \cdot \big(1 - S(\gamma\, x)/S(x)\big)$. The value $\mu(\mathcal{S})$ has been used before by Correa et al. [6] and Roughgarden [21] in the context of price of anarchy bounds for selfish routing where they showed that the routing cost of a Wardrop equilibrium is no more than a factor of $1/(1 - \mu(\mathcal{S}))$ away of the cost of a system optimum. For the special case that $\mathcal{S}$ is the set of polynomials with non-negative coefficients and maximal degree Δ, we derive exactly the approximation guarantees that Marcotte obtained for monomials. As an outcome of our more general analysis, we further derive that this algorithm is a 2-approximation for general convex latency functions and a 5/4-approximation for concave latency functions.

We then propose a new algorithm which we call ScaleUniformly. This algorithm first computes an optimal solution of the relaxation (as before) and then *uniformly* scales the capacities with a certain parameter $\lambda(\mathcal{S})$ that depends on the class of allowable latency functions $\mathcal{S}$. Based on well-known techniques using variational inequalities (Correa et al. [6] and Roughgarden [21]), we prove that this algorithm also yields a $(1 + \mu(\mathcal{S}))$-approximation. As our main result regarding approximation algorithms, we show that using the better of the two solutions returned by BringToEquilibrium and ScaleUniformly yields strictly better approximation guarantees. We give a closed form expression for the new approximation guarantee (as a function of $\mathcal{S}$) that, perhaps interestingly, depends not only on the well-known value $\mu(\mathcal{S})$ but also on the argument maximum $\gamma(\mathcal{S})$ in the definition of $\mu(\mathcal{S})$. We demonstrate the applicability of this general bound by showing that it achieves a 9/5-approximation for $\mathcal{S}$ containing arbitrary convex latencies. For affine latencies it achieves a $49/41 \approx 1.195$-approximation improving on the 5/4 of Marcotte. An overview of our results compared to those of Marcotte can be found in Table 1.

Some proofs missing in this extended abstract can be found in the full version.

1.2 Further Application

One of the most prominent and popular functions used in actual traffic models are the ones put forward by the Bureau of Public Roads (BPR) [25]. They are of the form $S_e(v_e) = t_e \cdot \left(1 + b_e \cdot (v_e/z_e)^4\right)$, where v_e is the edge flow, t_e represents the free-flow travel time, $b_e > 0$ is an edge-specific bias, and z_e represents the street capacity, e. g., in terms of the number of lanes and their width. Our best-of-two approximation algorithm yields an improved approximation factor for functions of this type (cf. Table 1 in the appendix) and can be employed to design road networks with a good tradeoff between construction cost and travel times.

Our results have impact beyond this classical application of designing street capacities. Also in telecommunication networks, Wardrop equilibria appear both in systems with source-routing as end-users choose least-delay paths, and in systems with distributed delay-based routing protocols such as OSPF when using the delay for setting the routing weights [26]. The latency at switches and routers depends on the installed capacity and has been modeled by functions of the form $S_e(v_e/z_e) = \rho\,(1+0.15\,(v_e/z_e))^4$, where ρ represents the propagation delay and z_e the installed capacity [20]. These functions fit into our framework, and our analysis improves the state-of-the-art to a 1.418-approximation and can be applied in scenarios where entire new networks have to be designed from scratch, e. g., after introducing new technology such as optical fiber cables. Our 9/5-approximation also applies to Davidson latency functions of the form $S_e(\frac{v_e}{z_e}) = \frac{v_e}{z_e}/(1 - \frac{v_e}{z_e}) = v_e/(z_e - v_e)$, where z_e represents the capacity of edge e.

1.3 Further Related Work

Quoting [27], CNDP has been recognized to be "one of the most difficult and challenging problems in transport" and there are numerous works approaching this problem. In light of the substantial literature on heuristics for CNDP, we refer the reader to the survey papers [5, 10, 17, 27].

While to the best of our knowledge prior to this work, the complexity status of CNDP was open, there have been several papers on the complexity of the *discrete (bilevel) network design problem*, DNDP for short, see [15, 22]. Given a network with edge latency functions and traffic demands, a basic variant of DNDP is to decide which edges should be removed from the network to obtain a Wardrop equilibrium in the resulting sub-network with minimum total travel time. This variant is motivated by the classical Braess paradox, where removing an edge from the network may improve the travel time of the new Wardrop equilibrium. Roughgarden [22] showed that DNDP is strongly NP-hard and that there is no $(\lfloor n/2 \rfloor - \epsilon)$-approximation algorithm (unless P = NP), even for single-commodity instances. He further showed that for single-commodity instances the trivial algorithm of not removing any edge from the graph is essentially best possible and achieves a $\lfloor n/2 \rfloor$-approximation. For affine latency functions, the trivial algorithm gives a 4/3-approximation (even for general networks) and this is also shown to be best possible. These results in comparison to ours highlight interesting differences. While DNDP is not approximable by any constant for convex latencies, for CNDP we give a 9/5-approximation. Moreover, all hardness results for DNDP already hold for single-commodity instances, while for CDNP we show that this case is solvable in polynomial time.

In independent work, Bhaskar et al. [4] studied a variant of CNDP where initial edge capacities are given and additional budget must be distributed among the edges to improve the resulting equilibrium. Among other results they show that the problem is NP-complete

in single-commodity networks that consist of parallel links in series. This again stands in contrast to our polynomial-time algorithm for CDNP for these instances.

2 Preliminaries

Let $G = (V, E)$ be a directed or undirected graph, V its set of vertices and $E \subseteq V \times V$ its set of edges. We are given a set K of *commodities*, where each commodity k is associated with a triple $(s_k, t_k, d_k) \in V \times V \times \mathbb{R}_{>0}$, where $s_k \in V$ is the *source*, $t_k \in V$ the *sink* and d_k the *demand* of commodity k. A multi-commodity flow on G is a collection of non-negative flow vectors $(\boldsymbol{v}^k)_{k\in K}$ such that for each $k \in K$ the flow vector $\boldsymbol{v}^k = (v_e^k)_{e\in E}$ satisfies the flow conservation constraints $\sum_{u\in V:(s_k,u)\in E} v^k_{s_k,u} = \sum_{u\in V:(u,t_k)\in E} v^k_{u,t_k} = d_k$ and $\sum_{u\in V:(u,w)\in E} v^k_{(u,w)} - \sum_{u\in V:(w,u)\in E} v^k_{(w,u)} = 0$ for all $w \in V \setminus \{s_k, t_k\}$. Whenever we write $\boldsymbol{v}$ without a superscript k for the commodity, we implicitly sum over all commodities, i. e., $v_e = \sum_{k\in K} v_e^k$ and $\boldsymbol{v} = (v_e)_{e\in E}$. We call v_e an *edge flow*. The set of all feasible edge flows will be denoted by $\mathcal{F}$.

The latency of each edge e depends on the installed capacity $z_e \geq 0$ and the edge flow v_e on e, and is given by a latency function $S_e : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ that maps v_e/z_e to a latency value $S_e(v_e/z_e)$, where we use the convention that $S_e(v_e/z_e) = \infty$ whenever $z_e = 0$. Throughout this paper, we assume that the set of allowable latency functions is restricted to some set $\mathcal{S}$ and we impose the following assumptions on $\mathcal{S}$.

▶ **Assumption 2.1.** *The set $\mathcal{S}$ of allowable latency functions only contains continuously differentiable and semi-convex functions S such that the functions $x \mapsto S(x)$ and $x \mapsto x^2 S'(x)$ are strictly increasing and unbounded.*

Assumption 2.1 is more general than requiring that all latency functions are stricly increasing and convex. For instance, the function $S(x) := \sqrt{x}$ satisfies Assumption 2.1 although it is concave.

Given a vector of capacities $\boldsymbol{z} = (z_e)_{e\in E}$, the latency of each edge e only depends on the edge flow v_e. Under these conditions, there exists a Wardrop flow $\boldsymbol{v} = (v_e)_{e\in E}$, i. e., a flow in which each commodity only uses paths of minimal latency. It is well known (cf. [3, 8, 24]) that each Wardrop flow is a solution to the optimization problem $\min_{\boldsymbol{v}\in\mathcal{F}} \sum_{e\in E} \int_0^{v_e} S_e(t/z_e)\,\mathrm{d}t$, and satisfies the *variational inequality*

$$\sum_{e\in E} S(v_e/z_e)(v_e - v'_e) \leq 0 \tag{2.1}$$

for every feasible flow $\boldsymbol{v}' \in \mathcal{F}$. For a vector of capacities $\boldsymbol{z}$ we denote by $\mathcal{W}(\boldsymbol{z})$ the corresponding set of Wardrop flows $\boldsymbol{v}(\boldsymbol{z})$. Beckmann et al. [3] showed that Wardrop flows and optimum flows are related:

▶ **Proposition 2.1** (Beckmann et al. [3])**.** *Let $S_e^*(x) = (xS_e(x))' = S_e(x) + xS'_e(x)$ be the* marginal cost function *of edge $e \in E$. Then $\boldsymbol{v}^*$ is an optimum flow with respect to the latency functions $(S_e)_{e\in E}$ if and only if it is Wardrop flow with respect to $(S_e^*)_{e\in E}$.*

In the continuous (bilevel) network design problem (CNDP) the goal is to buy capacities z_e at a price per unit $\ell_e > 0$ so as to minimize the sum of the construction cost $C^Z(\boldsymbol{v}, \boldsymbol{z}) = \sum_{e\in E} z_e\,\ell_e$ and the routing cost $C^R(\boldsymbol{v}, \boldsymbol{z}) = \sum_{e\in E} S_e(v_e/z_e)\,v_e$ of a resulting Wardrop equilibrium $\boldsymbol{v}$. Observe that $C^R(\boldsymbol{v}, \boldsymbol{z})$ is well defined as, by (2.1), it is the same for all Wardrop equilibria with respect to $\boldsymbol{z}$. Denote the combined cost by $C(\boldsymbol{v}, \boldsymbol{z}) = C^R(\boldsymbol{v}, \boldsymbol{z}) + C^Z(\boldsymbol{v}, \boldsymbol{z})$. We would like to reiterate that other linear combinations can be handled by scaling the capacity prizes accordingly.

▶ **Definition 1** (Continuous network design problem (CNDP))**.** Given a directed graph $G = (V, E)$ and for each edge e a latency function S_e and a construction cost $\ell_e > 0$, the continuous network design problem (CNDP) is to determine a non-negative capacity vector $\boldsymbol{z} = (z_e)_{e \in E}$ that minimizes

$$\min_{\boldsymbol{z} \geq 0} \min_{\boldsymbol{v} \in \mathcal{W}(\boldsymbol{z})} \sum_{e \in E} \big(S_e(v_e/z_e)\, v_e + z_e\, \ell_e \big). \tag{CNDP}$$

Relaxing the condition that $\boldsymbol{v}$ is a Wardrop equilibrium in (CNDP), we obtain the following relaxation of the continuous network design problem:

$$\min_{\boldsymbol{z} \geq 0} \min_{\boldsymbol{v} \in \mathcal{F}} \sum_{e \in E} \big(S_e(v_e/z_e)\, v_e + z_e\, \ell_e \big). \tag{CNDP'}$$

Marcotte [18] showed that for convex and unbounded latency functions, the relaxed problem (CNDP') can be solved efficiently by performing $|K|$ independent shortest path computations on the graph G, one for each commodity $k \in K$. The following proposition slightly generalizes his result to arbitrary, not necessarily convex latency functions that satisfy Assumption 2.1.

▶ **Proposition 2.2** (Marcotte [18])**.** *The relaxation* (CNDP') *can be solved by performing* $|K|$ *shortest path computations in polynomial time.*

▶ Remark. To speak about polynomial algorithms and hardness, we need to specify how the instances of CNDP, in particular the latency functions, are encoded, cf. [2, 11, 22]. While our hardness results hold even if all functions are linear and given by their rational coefficients, for our approximation algorithms, we require that we can solve (symbolically) equations involving a latency function and its derivative, e. g., Equation (4.4). Without this assumption, we still obtain the claimed approximation guarantees within arbitrary precision by polynomial time algorithms.

3 Hardness

As the main result of this section, we show that CNDP is APX-hard both on directed and undirected networks and even for affine latency functions. The proof of this result is technically quite involved, and we first show the weaker result that CNDP on directed networks is NP-complete.

▶ **Theorem 2.** *The continuous network design problem (CNDP) on directed networks is* NP*-complete in the strong sense, even if all latency functions are affine.*

Proof. CNDP lies in NP as a vector of capacities $\boldsymbol{z}$ is a polynomial certificate. Given $\boldsymbol{z}$, we can compute in polynomial time a corresponding Wardrop equilibrium and the total cost $C(\boldsymbol{v}, \boldsymbol{z})$.

To show the NP-hardness of the problem, we reduce from 3-SAT. Let ϕ be a Boolean formula in conjunctive normal form. We denote the set of variables and clauses of ϕ with $V(\phi)$ and $K(\phi)$, respectively, and set $\nu = |V(\phi)|$ and $\kappa = |K(\phi)|$. The set $L(\phi)$ of literals of ϕ contains for each variable $x_i \in V(\phi)$ the positive literal x_i and the negative literal $\bar{x}_i$, i. e., $L(\phi) = \{x_i \in V(\phi)\} \cup \{\bar{x}_i : x_i \in V(\phi)\}$. In the following, we will associate clauses with the set of literals that they contain.

We now explain the construction of a continuous network design problem based on ϕ that has the property that, for some $\epsilon \in (0, 1/8)$, an optimal solution has total cost less or equal

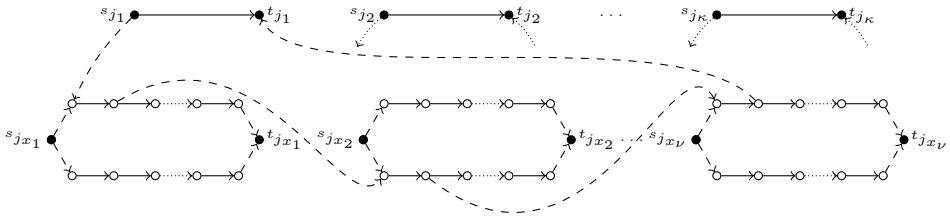


Figure 1 Network used to show the hardness of the continuous network design problem. Clause 1 is equal to $x_1 \vee \bar{x}_2 \vee x_\nu$. Dashed edges have zero latency.

to $(4+\epsilon)\kappa + 2\kappa\nu$ if and only if ϕ has a solution. Let $\epsilon \in (0, 1/8)$ be arbitrary. For each clause $k \in K(\phi)$, we introduce a *clause edge* e_k with latency function $S_{e_k}(v_{e_k}/z_{e_k}) = 4 + v_{e_k}/z_{e_k}$ and construction cost $\ell_{e_k} = (\epsilon/2)^2$. For each literal $l \in L(\phi)$ and each clause $k \in K(\phi)$, we introduce a *literal edge* $e_{l,k}$ with latency function $S_{e_{l,k}}(v_{e_{l,k}}/z_{e_{l,k}}) = v_{e_{l,k}}/z_{e_{l,k}}$ and cost $\ell_{e_{l,k}} = 1$. We denote the set of clause edges and literal edges by E_K and E_L, respectively.

For each variable $x_i \in V(\phi)$, there is a *variable commodity* j_{x_i} with source $s_{j_{x_i}}$, sink $t_{j_{x_i}}$ and demand $d_{j_{x_i}} = 1$. This commodity has two feasible paths, one path uses exclusively the literal edges $\{e_{x_i,k} : k \in K(\phi)\}$ that correspond to the non-negated variable x_i, the correspond to the negated variable $\bar{x}_i$. In that way, each feasible path of the variable commodity j_{x_i} corresponds to a `true`/`false` assignment of the variable x_i. For each clause $k = l_k \vee l'_k \vee l''_k$, we introduce a *clause commodity* j_k with source s_{j_k}, sink t_{j_k} and demand $d_{j_k} = 1$. The clause commodity may either choose its corresponding clause edge e_k or the corresponding literal edges that occur in k, i. e., $e_{l_k,k}$, $e_{l'_k,k}$, and $e_{l''_k,k}$. For notational convenience, we set $E_k = \{e_{l_k,k}, e_{l'_k,k}, e_{l''_k,k}\}$. We add some additional edges with latency 0 to obtain a network; see Fig. 1 where these edges are dashed. Note that the problem remains NP-hard, even if we do not allow edges with zero latency, see the full version of this paper.

First, we show that an optimal solution of the so-defined instance of the continuous network design problem P has total cost less or equal to $(4+\epsilon)\kappa + 2\kappa\nu$, if ϕ has a solution. To this end, let $\boldsymbol{y} = (y_{x_i})_{x_i \in V(\phi)}$ be a solution of ϕ. Then, a feasible solution of P is as follows: For each *positive* literal x_i that is selected in the solution y_i, we buy capacity 1 for the corresponding *negative* literal edges $\{e_{\bar{x}_i,k} : k \in K(\phi)\}$, and vice versa. Formally, we set

$$z_{a_{l,k}} = \begin{cases} 1, & \text{if } l = x_i \text{ and } y_{x_i} = \texttt{false}, \\ 1, & \text{if } l = \bar{x}_i \text{ and } y_{x_i} = \texttt{true}, \\ 0, & \text{otherwise.} \end{cases}$$

For each clause edge e_k, $k \in K(\phi)$, we buy capacity $2/\epsilon$. This particular capacity vector $\boldsymbol{z} = (z_e)_{e \in E}$ implies that each variable commodity j_{x_i} has a unique path of finite length, i. e., the path using the edges corresponding to the *negation* of the corresponding literal in $\boldsymbol{y}$. Using that $\boldsymbol{y}$ is a solution of ϕ, we further obtain that for each clause commodity j_k at least one of the edges in E_k has capacity zero and, thus, infinite latency. This implies that, in the unique Wardrop equilibrium, the demand of each clause commodity j_k is routed along the corresponding clause edge e_k. For the total cost of this solution, we obtain

$$\begin{aligned} C(\boldsymbol{v}, \boldsymbol{z}) &= \sum_{e \in E_K} \big((4 + v_e/z_e)v_e + (\epsilon/2)^2 z_e\big) + \sum_{e \in E_L} \big((v_e/z_e)v_e + z_e\big) \\ &= \sum_{e \in E_K} \big((4 + \epsilon/2) + (\epsilon/2)\big) + \frac{1}{2} \sum_{e \in E_L} \big(1 + 1\big) = (4+\epsilon)\kappa + 2\kappa\nu. \end{aligned} \tag{3.1}$$

Hence, an optimal solution has cost not larger than (3.1) if ϕ has a solution.

We proceed to prove that the total cost of an optimal solution are strictly larger than (3.1) if ϕ does *not* admit a solution. Let $\boldsymbol{z} = (z_e)_{e\in E}$ be an optimal solution of P and let $\boldsymbol{v} = (v_e)_{e\in E}$ be a corresponding Wardrop flow. We distinguish two cases.

First case: $v_{e_k} > 0$ for all $k \in K(\phi)$, i. e., each clause commodity j_k sends flow over the corresponding clause edge e_k.

Before we prove the thesis for this case, we need some additional notation. For the Wardrop flow v_e on edge $e \in E$, let v_e^V and v_e^K denote the flow on e that is due to the variable commodities and the clause commodities, respectively. We claim that there is a clause $\tilde{k} \in K(\phi)$, $\tilde{k} = l_{\tilde{k}} \vee l'_{\tilde{k}} \vee l''_{\tilde{k}}$ such that the flow of the variable commodities on each of the corresponding literal edges in $E_{\tilde{k}} = \{e_{l_{\tilde{k}},\tilde{k}}, e_{l'_{\tilde{k}},\tilde{k}}, e_{l''_{\tilde{k}},\tilde{k}}\}$ is at least $1/2$, i. e.,

$$v^V_{e_{l_{\tilde{k}},\tilde{k}}} \geq 1/2, \qquad v^V_{e_{l'_{\tilde{k}},\tilde{k}}} \geq 1/2, \qquad \text{and} \qquad v^V_{e_{l''_{\tilde{k}},\tilde{k}}} \geq 1/2. \tag{3.2}$$

For a contradiction, let us assume that for each clause $k = l_k \vee l'_k \vee k''_k$ there is a literal $l^*_k \in \{l_k, l'_k, l''_k\}$ such that $v^V_{e_{l^*_k,k}} < 1/2$. As each variable $x_i \in V(\phi)$ splits its unit demand between the path consisting of the positive literal edges $\{e_{x_i,k} : k \in K(\phi)\}$ and the path consisting of the negative literal edges $\{e_{\bar{x}_i,k} : k \in K(\phi)\}$, at most one of these two paths is used with a flow strictly smaller than $1/2$. Thus, the assignment vector $\boldsymbol{y}$ defined as

$$y_{x_i} = \begin{cases} \texttt{true}, & \text{if } v^V_e < 1/2 \text{ for all } e \in \{e_{x_i,k} : k \in K(\phi)\}, \\ \texttt{false}, & \text{if } v^V_e < 1/2 \text{ for all } e \in \{e_{\bar{x}_i,k} : k \in K(\phi)\}, \\ \texttt{true}, & \text{otherwise,} \end{cases}$$

is well-defined. By construction, $\boldsymbol{y}$ satisfies all clauses, which is a contradiction to the assumption that no such assignment exists. We conclude that there is a clause $\tilde{k}$ such that (3.2) holds.

We proceed to bound the total cost of a solution. As $\boldsymbol{v}$ is a Wardrop equilibrium in which the clause commodity $j_{\tilde{k}}$ uses at least partially the clause edge $e_{\tilde{k}}$, we further derive that $\sum_{e\in E_{\tilde{k}}} v_e/z_e \geq v_{e_{\tilde{k}}}/z_{e_{\tilde{k}}} > 4$. We bound the total cost of the solution $(\boldsymbol{v}, \boldsymbol{z})$ by observing

$$\begin{aligned} C(\boldsymbol{v},\boldsymbol{z}) &= \sum_{e\in E_L} \left(v_e^2/z_e + z_e\right) + \sum_{e\in E_K} \left((4 + v_e/z_e)v_e + (\epsilon/2)^2 z_e\right) \\ &\geq \sum_{e\in E_L} \overline{\min}_{z_e\geq 0}\left(v_e^2/z_e + z_e\right) + \sum_{e\in E_K} \overline{\min}_{z_e\geq 0}\left((4 + v_e/z_e)v_e + (\epsilon/2)^2 z_e\right), \end{aligned}$$

where we slightly abuse notation by writing $\overline{\min}_{z_e\geq 0}$ shorthand for $\min_{z_e\geq 0:\boldsymbol{v}\in\mathcal{W}(z)}$. We obtain an upper bound by relaxing $\overline{\min}_{z_e\geq 0}$ to $\min_{z_e\geq 0}$ for the edges in $E_L \setminus E_{\tilde{k}}$ and E_K. Hence,

$$\begin{aligned} C(\boldsymbol{v},\boldsymbol{z}) \geq &\sum_{e\in E_L\setminus E_{\tilde{k}}} \min_{z_e\geq 0}\left(v_e^2/z_e + z_e\right) + \sum_{e\in E_{\tilde{k}}} \overline{\min}_{z_e\geq 0}\left(v_e^2/z_e + z_e\right) \\ &+ \sum_{e\in E_K} \min_{z_e\geq 0}\left((4 + v_e/z_e)v_e + (\epsilon/2)^2 z_e\right). \end{aligned}$$

Calculating the respective minima, we obtain

$$C(\boldsymbol{v},\boldsymbol{z}) \geq \sum_{e\in E_L\setminus E_{\tilde{k}}} 2v_e + \sum_{e\in E_{\tilde{k}}} \underbrace{\overline{\min}_{z_e\geq 0}\left(v_e^2/z_e + z_e\right)}_{\geq 2v_e} + \sum_{e\in E_K} (4+\epsilon)v_e. \tag{3.3}$$

Each clause commodity j_k can route its demand either over the clause edge e_k or over the three literal edges in E_k. Every fraction of the demand routed over the clause edge contributes $4+\epsilon$ to the expression on the right hand side of (3.3) while it contributes at least 6 when routed over the literal edges. Thus, the right hand side of (3.3) is minimized when the clause commodities do not use the literal edges at all. We then obtain

$$\begin{aligned} C(\boldsymbol{v},\boldsymbol{z}) &\geq \sum_{e\in E_L\setminus E_{\tilde{k}}} 2v_e^V + \sum_{e\in E_{\tilde{k}}} \overline{\min_{z_e\geq 0}}\big((v_e^V)^2/z_e + z_e\big) + (4+\epsilon)|E_K| \\ &= 2\Big(\kappa\nu - \sum_{e\in E_{\tilde{k}}} v_e^V\Big) + (4+\epsilon)\kappa + \sum_{e\in E_k} \overline{\min_{z_e\geq 0}}\big((v_e^V)^2/z_e + z_e\big), \\ &= 2\kappa\nu + (4+\epsilon)\kappa + \sum_{e\in E_{\tilde{k}}} \overline{\min_{z_e\geq 0}}\big((v_e^V)^2/z_e + z_e - 2v_e^V\big), \\ &> 2\kappa\nu + (4+\epsilon)\kappa + Q, \end{aligned}$$

where Q is the solution to the constrained minimization problem

$$\begin{aligned} Q = \min_{\substack{v_e^V, z_e>0 \\ e\in E_{\tilde{k}}}} & \sum_{e\in E_{\tilde{k}}} \big((v_e^V)^2/z_e + z_e - 2v_e^V\big) \\ \text{s. t.:} \quad & \sum_{e\in E_{\tilde{k}}} v_e^V/z_e \geq 4 && (3.4) \\ & v_e^V \geq 1/2 \text{ for all } e\in E_{\tilde{k}}. && (3.5) \end{aligned}$$

Side constraint (3.4) is a relaxation of the requirement that $\boldsymbol{v}$ is a Wardrop equilibrium as the latency of the literal edges is strictly larger than 4. Side constraint (3.5) is due to the fact that for clause $\tilde{k}$ the three corresponding literal edges $e_{l_{\tilde{k}},\tilde{k}}$, $e_{l'_{\tilde{k}},\tilde{k}}$, and $e_{l''_{\tilde{k}},\tilde{k}}$ are used with a flow of at least $1/2$ by the variable commodities. The optimal solution to the constraint optimization problem Q is equal to $Q = 1/8$ and is attained for $v_e^V = 1/2$ and $z_e = 3/8$ for all $e\in E_{\tilde{k}}$. This implies that the total cost of a solution is not smaller than $(4+\epsilon)\kappa + 2\kappa\nu + 1/8$, which finishes the first case of this proof.

Second case: There is a clause commodity $j_{\tilde{k}}$ that does not use its clause edge $e_{\tilde{k}}$, i. e., $v_{e_{\tilde{k}}} = 0$. As for first case, we observe

$$C(\boldsymbol{v},\boldsymbol{z}) = \sum_{e\in E_L}\big(v_e^2/z_e + z_e\big) + \sum_{e\in E_K}(4v_e + v_e^2/z_e + (\epsilon/2)^2 z_e) \geq \sum_{e\in E_L} 2v_e + \sum_{e\in E_K}(4+\epsilon)v_e.$$

Using that $j_{\tilde{k}}$ does not use its clause edge, we derive that the flow on the literal edges amounts to $\nu\kappa + 3$ and we obtain

$$C(\boldsymbol{v},\boldsymbol{z}) \geq 2(\kappa\nu+3) + (4+\epsilon)(\kappa-1) = 2\kappa\nu + (4+\epsilon)\kappa + 2,$$

which concludes the proof. ◀

With a more involved construction and a more detailed analysis, we can show that CNDP is in fact APX-hard. For this proof, we use a similar construction as in the proof of Theorem 2 but reduce from a specific variant of MAX-3-SAT, which is NP-hard to approximate. Due to space constraints we defer the details to the full version of this paper.

▶ **Theorem 3.** *The continuous network design problem (CNDP) on directed networks is* APX*-hard, even if all latency function are affine.*

With a similar construction, we can also show APX-hardness for CNDP on undirected networks as well, see the full version of this paper. For our hardness results, we use instances with different sinks. In contrast, CNDP can be solved efficiently for networks with a single sink.

▶ **Proposition 3.1.** *In networks with only one sink vertex t, the continuous network design problem (CNDP) can be solved in polynomial time.*

4 Approximation

Given the APX-hardness of the problem, we study the approximation of CNDP. We first provide a detailed analysis of the approximation guarantees of two different approximation algorithms. Then, as the arguably most interesting result of this section, we provide an improved approximation guarantee for taking the better of the two algorithms. The approximation guarantees proven in this section depend on the set $\mathcal{S}$ of allowable cost functions and are in fact closely related to the *anarchy value* value $\alpha(\mathcal{S})$ introduced by Roughgarden [21] and Correa et al. [6]. Intuitively, the anarchy value of a set of latency functions $\mathcal{S}$ is the worst case ratio between the routing cost of a Wardrop equilibrium and that of a system optimum of an instance in which all latency functions are contained in $\mathcal{S}$. Roughgarden [21] and Correa et al. [6] show that $\alpha(\mathcal{S}) = 1/(1-\mu(\mathcal{S}))$, where

$$\mu(\mathcal{S}) = \sup_{S\in\mathcal{S}} \sup_{x\geq 0} \max_{\gamma\in[0,1]} \gamma\cdot\Big(1-\frac{S(\gamma\,x)}{S(x)}\Big). \tag{4.1}$$

For a set $\mathcal{S}$ of latency functions, we denote by $\gamma(\mathcal{S})$ the argmaximum γ in (4.1) for which $\mu(\mathcal{S})$ is achieved. The following lemma gives an alternative representation of $\mu(S)$.

▶ **Lemma 4.** *For a latency function S,*

$$\sup_{x\geq 0}\max_{\gamma\in[0,1]}\Big\{\gamma\Big(1-\frac{S(\gamma\,x)}{S(x)}\Big)\Big\} = \sup_{x\geq 0}\Big\{\gamma\cdot\frac{S'(x)\,x}{S(x)+S'(x)\,x} : S(x)+S'(x)\,x = S(x/\gamma)\Big\}.$$

Proof. The expression $\sup_{x\geq 0}\max_{\gamma\in[0,1]}\gamma\left(1-\frac{S(\gamma\,x)}{S(x)}\right)$ is non-negative and strictly positive for $\gamma\in(0,1)$, thus, the inner maximum is attained for $\gamma\in(0,1)$. Hence, γ satisfies the first order optimality conditions

$$\begin{aligned} & 0 = \Big(1-\frac{S(\gamma\,x)}{S(x)}\Big) - \gamma\,x\cdot\frac{S'(\gamma\,x)}{S(x)} \\ \Leftrightarrow\quad & S(x) = S(\gamma\,x) + \gamma\,x\,S'(\gamma\,x) \end{aligned}$$

By substituting $y=\gamma\,x$, we obtain

$$\begin{aligned} & \sup_{x\geq 0}\max_{\gamma\in[0,1]}\gamma\Big(1-\frac{S(\gamma\,x)}{S(x)}\Big) \\ & = \sup_{y\geq 0}\Big\{\gamma\Big(1-\frac{S(y)}{S(y/\gamma)}\Big) : \gamma\in[0,1] \text{ with } S(y/\gamma) = S(y)+S'(y)\,y\Big\} \\ & = \sup_{y\geq 0}\Big\{\gamma\cdot\frac{S'(y)\,y}{S(y)+S'(y)\,y} : \gamma\in[0,1] \text{ with } S(y/\gamma) = S(y)+S'(y)\,y\Big\}, \end{aligned}$$

which proves the lemma. ◀

Algorithm 1 BringToEquilibrium

1: $(\boldsymbol{v}^*, \boldsymbol{z}^*) \leftarrow$ solution to (CNDP').
2: **for all** $e \in E$ **do**
3: $\quad \delta_e \leftarrow v_e^*/z_e^*$
4: $\quad \gamma_e \leftarrow$ solution to $S_e(\delta_e) + S_e'(\delta_e)\delta_e = S_e(\frac{\delta_e}{\gamma_e})$
5: $\quad z_e \leftarrow \gamma_e z_e^*$
6: **end for**
7: **return** $(\boldsymbol{v}^*, \boldsymbol{z})$

Algorithm 2 ScaleUniformly

1: $(\boldsymbol{v}^*, \boldsymbol{z}^*) \leftarrow$ solution to (CNDP').
2: $p \leftarrow C^R(\boldsymbol{v}^*, \boldsymbol{z}^*)/C(\boldsymbol{v}^*, \boldsymbol{z}^*)$
3: $\lambda \leftarrow \mu(\mathcal{S}) + \sqrt{\mu(\mathcal{S})\frac{p}{1-p}}$
4: Compute Wardrop equilibrium $\boldsymbol{v}$ with respect to scaled capacities $\lambda \boldsymbol{z}^*$.
5: **return** $(\boldsymbol{v}, \lambda \boldsymbol{z}^*)$

4.1 Two Approximation Algorithms

The first algorithm that we call BringToEquilibrium (cf. Algorithm 1) was already proposed by Marcotte [18, Section 4.3] and analyzed for monomial latency functions. Our contribution is a more general analysis of BringToEquilibrium that works for arbitrary sets of latency functions $\mathcal{S}$, requiring only Assumption 2.1. The second algorithm, that we call ScaleUniformly (cf. Algorithm 2), is a new algorithm that we introduce in this paper.

For both approximation algorithms, we first compute an optimum solution $(\boldsymbol{v}^*, \boldsymbol{z}^*)$ to a relaxation of CNDP without the equilibrium constraints, i. e., we compute a solution $(\boldsymbol{v}^*, \boldsymbol{z}^*)$ to the problem $\min_{\boldsymbol{z} \geq 0} \min_{v \in \mathcal{F}} \sum_{e \in E} \big(S_e(v_e/z_e)\, v_e + z_e\, \ell_e\big)$, which can be done in polynomial time (Proposition 2.2). Then, in both algorithms, we reduce the capacity vector $\boldsymbol{z}^*$, and determine a Wardrop equilibrium for the new capacity vector. The algorithms differ in the way we adjust the capacity vector $\boldsymbol{z}^*$. While in BringToEquilibrium, we reduce the edge capacities individually such that the optimum solution to the relaxation (CNDP') is a Wardrop equilibrium, in ScaleUniformly, we scale all capacities uniformly by a factor λ (cf. line 2-3) and compute a Wardrop equilibrium for the scaled capacities.

We first show that the approximation guarantee of BringToEquilibrium is at most $(1 + \mu(\mathcal{S}))$. For the proof of this result, we use the first order optimality conditions for the vector of capacities $\boldsymbol{v}^*$ obtained as a solution to the relaxed problem (CNDP') in combination with the variational inequalities technique used in the price of anarchy literature (e. g. Roughgarden [21] and Correa et al. [6]).

▶ **Theorem 5.** *The approximation guarantee of* BringToEquilibrium *is at most* $1 + \mu(\mathcal{S})$.

Proof. Let $(\boldsymbol{v}^*, \boldsymbol{z}^*)$ be the relaxed solution computed in the first step of BringToEquilibrium. By the necessary Karush-Kuhn-Tucker optimality conditions, $(\boldsymbol{v}^*, \boldsymbol{z}^*)$ satisfies

$$\ell_e = S_e'(v_e^*/z_e^*)(v_e^*/z_e^*)^2, \text{ for all } e \in E \text{ with } z_e^* > 0\,. \tag{4.2}$$

Eliminating ℓ_e in the statement of the relaxed problem (CNDP') we obtain the following expression for the total cost of the relaxation:

$$C(\boldsymbol{v^*}, \boldsymbol{z^*}) = \sum_{e \in E} \Big(S_e(v_e^*/z_e^*) + S_e'(v_e^*/z_e^*)(v_e^*/z_e^*)\Big) v_e^*\,. \tag{4.3}$$

For each $e \in E$ let $\delta_e = v_e^*/z_e^*$, if $z_e^* > 0$, and $\delta_e = 0$, otherwise. We define a new vector of capacities $\boldsymbol{z}$ by $z_e = \gamma_e \cdot z_e^*, e \in E$, where $\gamma_e \in [0, 1]$ is a solution to the equation

$$S_e(\delta_e) + S_e'(\delta_e)\, \delta_e = S_e(\delta_e/\gamma_e). \tag{4.4}$$

By Proposition 2.1, the flow $\boldsymbol{v}^*$ is a Wardrop flow with respect to $\boldsymbol{z}$. We are interested in

bounding $C(\boldsymbol{v}^*, \boldsymbol{z})$. To this end, we calculate

$$C(\boldsymbol{v}^*, \boldsymbol{z}) = \sum_{e\in E} (S_e(\delta_e/\gamma_e)v_e^* + \ell_e\, z_e) \overset{(4.4)}{=} \sum_{e\in E}\Big(\big(S_e(\delta_e) + S_e'(\delta_e)\,\delta_e\big)v_e^* + \gamma_e\,\ell_e\, z_e^*\Big)$$
$$\overset{(4.2)}{=} \sum_{e\in E}\Big(\big(S_e(\delta_e) + S_e'(\delta_e)\,\delta_e\big)v_e^* + \gamma_e\, S_e'(\delta_e)\,\delta_e\, v_e^*\Big). \tag{4.5}$$

By (4.1),(4.4), and Lemma 4, we have $\gamma_e\, S_e'(\delta_e)\,\delta_e \le \mu(\mathcal{S})\,(S_e(\delta_e) + S_e'(\delta_e)\,\delta_e)$. Combining this inequality with (4.5), gives

$$C(\boldsymbol{v}^*, \boldsymbol{z}) \le (1+\mu(\mathcal{S}))\sum_{e\in E}\Big(\big(S_e(\delta_e) + S_e'(\delta_e)\,\delta_e\big)v_e^* \overset{(4.3)}{=} (1+\mu(\mathcal{S}))\,C(\boldsymbol{v}^*, \boldsymbol{z}^*),$$

which completes the proof of the theorem. ◀

We proceed by showing that ScaleUniformly achieves the same approximation guarantee of $1+\mu(\mathcal{S})$. Recall that ScaleUniformly first computes a relaxed solution $(\boldsymbol{v}^*, z^*)$. Then, this relaxed solution is used to compute an optimal scaling factor $\lambda \le 1$ with which all capacities are scaled subsequently. The algorithm then returns the scaled capacity vector $\lambda\boldsymbol{z}^*$ together with a corresponding Wardrop equilibrium $v \in \mathcal{W}(\lambda\boldsymbol{z}^*)$.

An (worse) approximation guarantee of 2 can be inferred directly from a bicriteria result of Roughgarden and Tardos [23] who showed that for any instance the routing cost of a Wardrop equilibrium is not worse than a system optimum that ships twice as much flow. This implies that for $\lambda = 1/2$ we have $C(v, \lambda z^*) \le 2C(v^*, z^*)$, as claimed.

For the proof of the following result, we take a different road that allows us to express the approximation guarantee of ScaleUniformly as a function of the parameter p defined as the fraction of the total cost $C(\boldsymbol{v}^*, \boldsymbol{z}^*)$ of the relaxed solution allotted to the routing costs $C^R(\boldsymbol{v}^*, \boldsymbol{z}^*)$. This is an important ingredient for the analysis of the best-of-two algorithm.

▶ **Theorem 6.** *The approximation guarantee of* ScaleUniformly *is at most* $(1+\mu(\mathcal{S}))$.

Proof. The algorithm first computes an optimum solution $(\boldsymbol{v}^*, \boldsymbol{z}^*)$ of the relaxed problem (CNDP'). Then $p \in [0,1]$ is defined as the fraction of $C(\boldsymbol{v}^*, \boldsymbol{z}^*)$ that corresponds to the routing cost $C^R(\boldsymbol{v}^*, \boldsymbol{z}^*)$, i. e., $C^R(\boldsymbol{v}^*, \boldsymbol{z}^*) = \sum_{e\in E} S_e(v_e^*/z_e^*)\, v_e^* = p\,C(\boldsymbol{v}^*, \boldsymbol{z}^*)$. Now, we define $\lambda = \mu(\mathcal{S}) + \sqrt{\mu(\mathcal{S})\frac{p}{1-p}}$ and consider the capacity vector $\lambda\boldsymbol{z}^*$, in which the capacities of the optimal solution to the relaxation are scaled uniformly by λ. Finally, we compute a Wardrop equilibrium with respect to capacities $\lambda\boldsymbol{z}^*$. Let $\boldsymbol{v}$ the corresponding equilibrium flow. We now bound the routing and installation cost of $(\boldsymbol{v}, \lambda\boldsymbol{z}^*)$ separately. For the installation cost, we obtain

$$C^Z(\boldsymbol{v}, \lambda\boldsymbol{z}^*) = \sum_{e\in E} \lambda\,\ell_e\, z_e = \lambda(1-p)\,C(\boldsymbol{v}^*, \boldsymbol{z})$$

and for the routing cost

$$C^R(\boldsymbol{v}, \lambda\boldsymbol{z}^*) = \sum_{e\in E} S_e\Big(\frac{v_e}{\lambda z_e^*}\Big)\, v_e \le \sum_{e\in E} S_e\Big(\frac{v_e}{\lambda z_e^*}\Big)\, v_e^*$$
$$= p\,C(\boldsymbol{v}^*, \boldsymbol{z}^*) + \sum_{e\in E}\left(S_e\Big(\frac{v_e}{\lambda z_e^*}\Big)\, v_e^* - S_e\Big(\frac{v_e^*}{z_e^*}\Big)\, v_e^*\right), \tag{4.6}$$

where the first inequality uses the variational inequality (2.1). We proceed to bound $S_e(\frac{v_e}{\lambda z_e^*})\, v_e^* - S_e(\frac{v_e^*}{z_e^*})\, v_e^*$ in terms of the routing cost $S_e(\frac{v_e}{\lambda z_e^*})\, v_e$ for that edge e. To this end,

note that for each edge $e \in E$ we have

$$\frac{S_e(\frac{v_e}{\lambda z_e^*})v_e^* - S_e\big(\frac{v_e^*}{z_e^*}\big)v_e^*}{S_e(\frac{v_e}{\lambda z_e^*})v_e} \leq \sup_{S\in\mathcal{S}} \sup_{x,y,z\geq 0} \frac{S(\frac{y}{\lambda z})x - S(\frac{x}{z})x}{S(\frac{y}{\lambda z})y} \tag{4.7}$$

$$= \sup_{S\in\mathcal{S}} \sup_{x,y\geq 0} \frac{S(\frac{y}{\lambda})x - S(x)x}{S(\frac{y}{\lambda})y} = \sup_{S\in\mathcal{S}} \sup_{x,y\geq 0} \frac{S(y)\,x - S(x)x}{S(y)\lambda y}. \tag{4.8}$$

This implies $y \geq x$ and we may substitute $x = \gamma\, y$ with $\gamma \in [0,1]$. We then obtain for each edge $e \in E$ that

$$\frac{S_e(\frac{v_e}{\lambda z_e^*})v_e^* - S_e\big(\frac{v_e^*}{z_e^*}\big)v_e^*}{S_e(\frac{v_e}{\lambda z_e^*})v_e} \leq \sup_{S\in\mathcal{S}} \sup_{y\geq 0} \max_{\gamma\in[0,1]} \frac{\gamma S(y) - \gamma S(\gamma\, y)}{\lambda\, S(y)} \tag{4.9}$$

$$= \sup_{S\in\mathcal{S}} \sup_{y\geq 0} \max_{\gamma\in[0,1]} \frac{\gamma}{\lambda}\Big(1 - \frac{S(\gamma\, y)}{S(y)}\Big) = \frac{\mu(\mathcal{S})}{\lambda}. \tag{4.10}$$

Combining (4.10) and (4.6), we obtain $C^R(\boldsymbol{v}, \lambda\boldsymbol{z}^*) \leq p\, C(\boldsymbol{v}^*, \boldsymbol{z}^*) + \frac{\mu(\mathcal{S})}{\lambda}\, C^R(\boldsymbol{v}, \lambda\boldsymbol{z}^*)$ or, equivalently, $C^R(\boldsymbol{v}, \lambda\boldsymbol{z}^*) \leq \frac{p}{1-\mu(\mathcal{S})/\lambda} C(\boldsymbol{v}^*, \boldsymbol{z}^*)$. Thus, we can bound the total cost of the outcome of ScaleUniformly by

$$C(\boldsymbol{v}, \lambda\boldsymbol{z}^*) = C^R(\boldsymbol{v}, \lambda\boldsymbol{z}^*) + C^Z(\boldsymbol{v}, \lambda\boldsymbol{z}^*) \leq \frac{p}{1-\mu(\mathcal{S})/\lambda} C(\boldsymbol{v}^*, \boldsymbol{z}^*) + \lambda(1-p)\, C(\boldsymbol{v}^*, \boldsymbol{z}^*)$$
$$= \lambda\Big(\frac{p}{\lambda - \mu(\mathcal{S})} + 1 - p\Big) C(\boldsymbol{v}^*, \boldsymbol{z}^*).$$

Since $\lambda = \mu(\mathcal{S}) + \sqrt{\mu(\mathcal{S})\frac{p}{1-p}}$, we obtain

$$\frac{C(\boldsymbol{v}, \lambda\boldsymbol{z}^*)}{C(\boldsymbol{v}^*, \boldsymbol{z}^*)} \leq p + 2\sqrt{p(1-p)\mu(\mathcal{S})} + \mu(\mathcal{S})(1-p) = \big(\sqrt{p} + \sqrt{\mu(\mathcal{S})(1-p)}\big)^2. \tag{4.11}$$

Elementary calculus shows that $\big(\sqrt{p} + \sqrt{\mu(\mathcal{S})(1-p)}\big)^2$ attains its maximum at $p = \frac{1}{1+\mu(S)}$. Substituting this value into (4.11) gives $C(\boldsymbol{v}, \lambda\boldsymbol{z}^*)/C(\boldsymbol{v}^*, \boldsymbol{z}^*) \leq 1 + \mu(\mathcal{S})$, as claimed. ◀

For particular sets $\mathcal{S}$ of latency functions, we compute upper bounds on $\mu(\mathcal{S})$ in order to obtain an explicit upper bound on the approximation guarantees of BringToEquilibrium and ScaleUniformly. We then obtain the following corollary of Theorem 5 and Theorem 6.

▶ **Corollary 7.** *For a set $\mathcal{S}$ of latency functions satisfying Assumption 2.1, the approximation guarantee of* BringToEquilibrium *and* ScaleUniformly *is at most*

(a) 2*, without further requirements on $\mathcal{S}$.*

(b) 5/4*, if $\mathcal{S}$ contains concave latencies only,*

(c) $1 + \frac{\Delta}{\Delta+1}\big(\frac{1}{\Delta+1}\big)^{1/\Delta}$*, if $\mathcal{S}$ contains only polynomials with non-negative coefficients and degree at most Δ, i. e., each $S \in \mathcal{S}$ is of the form $S(x) = \sum_{j=0}^{\Delta} a_j x^j$ with $a_j \geq 0$ for all j.*

4.2 Best-of-Two Approximation

In this section we show that although both BringToEquilibrium and ScaleUniformly achieve an approximation guarantee of $(1 + \mu(\mathcal{S}))$ taking the better of the two algorithms we obtain a strictly better performance guarantee.

The key idea of the proof is to extend the analysis of the BringToEquilibrium algorithm in order to express its approximation guarantee as a function of the parameter p that measures the proportion of the routing cost in the total cost of a relaxed solution. This allows us to determine the worst-case p for which the approximation guarantee of the both algorithm is maximized.

▶ **Theorem 8.** *Taking the better solution of* BringToEquilibrium *and* ScaleUniformly *has an approximation guarantee of at most* $\frac{(\gamma(\mathcal{S})+\mu(\mathcal{S})+1)^2}{(\gamma(\mathcal{S})+\mu(\mathcal{S})+1)^2-4\mu(\mathcal{S})\gamma(\mathcal{S})}$*, which is strictly smaller than* $1+\mu(\mathcal{S})$*.*

Proof. Recall from (4.11) that the approximation guarantee of the algorithm ScaleUniformly is

$$\left(\sqrt{p}+\sqrt{\mu(\mathcal{S})(1-p)}\right)^2,$$

where $p = C^R(\boldsymbol{v}^*,\boldsymbol{z}^*)/C(\boldsymbol{v}^*,\boldsymbol{z}^*)$. We extend our analysis of BringToEquilibrium using this parameter p. With the notation in Theorem 5, by (4.5), BringToEquilibrium returns a feasible solution $(\boldsymbol{v}^*,\boldsymbol{z})$ with

$$\begin{aligned}
C(\boldsymbol{v}^*,\boldsymbol{z}) &= \sum_{e\in E}\Big(\big(S_e(\delta_e)+S_e'(\delta_e)\,\delta_e\big)v_e^* + \gamma_e\, S_e'(\delta_e)\,\delta_e\, v_e^*\Big)\\
&= p\,C(\boldsymbol{v}^*,\boldsymbol{z}^*) + \sum_{e\in E} S_e'(\delta_e)\,\delta_e\, v_e^*(1+\gamma_e)\\
&\le p\,C(\boldsymbol{v}^*,\boldsymbol{z}^*) + (1+\gamma(\mathcal{S}))\sum_{e\in E} S_e'(\delta_e)\,\delta_e\, v_e^*\\
&= p\,C(\boldsymbol{v}^*,\boldsymbol{z}^*) + (1+\gamma(\mathcal{S}))(1-p)\,C(\boldsymbol{v}^*,\boldsymbol{z}^*)\\
&= \big(1+\gamma(\mathcal{S})(1-p)\big)\,C(\boldsymbol{v}^*,\boldsymbol{z}^*).
\end{aligned}$$

Thus, by taking the best of the two heuristics, we obtain an approximation guarantee of

$$\max_{p\in(0,1)} \min\left\{1+\gamma(\mathcal{S})(1-p), \left(\sqrt{p}+\sqrt{\mu(\mathcal{S})(1-p)}\right)^2\right\}.$$

The maximum of this expression is attained for

$$p = p^* := \frac{(\gamma(\mathcal{S})-\mu(\mathcal{S})+1)^2}{(\gamma(\mathcal{S})-\mu(\mathcal{S})+1)^2+4\mu(\mathcal{S})} \tag{4.12}$$

which yields the claimed improved upper bound. ◀

It is not necessary to run both approximation algorithms to get this approximation guarantee. After computing the optimum solution to the relaxation (CNDP'), we can determine the value for $p = C^R(\boldsymbol{v}^*,\boldsymbol{z}^*)/C(\boldsymbol{v}^*,\boldsymbol{z}^*)$ and proceed with ScaleUniformly if $p \le p^*$ (cf. (4.12)) and with BringToEquilibrium otherwise.

For particular sets $\mathcal{S}$ of latency functions, we evaluate $\mu(\mathcal{S})$ and $\gamma(\mathcal{S})$ and obtain the following corollary of Theorem 8.

▶ **Corollary 9.** *For a set* $\mathcal{S}$ *of latency functions satisfying Assumption 2.1, the approximation guarantee in Theorem 8 is at most*

- **(a)** $9/5$*, without further requirements on* $\mathcal{S}$*,*
- **(b)** $49/41 \approx 1.195$*, if* $\mathcal{S}$ *contains concave latencies only.*
- **(c)** $1+\frac{4\Delta(\Delta+1)}{2(2\Delta+1)(\Delta+1)^{1+1/\Delta}+(\Delta+1)^{2(1+1/\Delta)}+1}$*, if* $\mathcal{S}$ *contains only polynomials with non-negative coefficients and degree at most* Δ*, i. e., every* $S\in\mathcal{S}$ *is of the form* $S(x)=\sum_{j=0}^{\Delta} a_j x^j$ *with* $a_j \ge 0$ *for all* j*.*

5 Conclusion

We reconsidered the classical continuous network design problem (CNDP) and established the first hardness result for CNDP. Further, we provided a general approximation guarantee for an algorithm studied by Marcotte [18] depending on the set of allowed cost functions which is related to the *anarchy value* of the set of cost functions. We then showed that the approximation of the problem can improved by taking the best of that algorithm and another approximation algorithm that we propose.

In the transportation literature, further variants of CNDP have been investigated. One such example are situations in which the network designer is only interested in minimizing total travel time but investments are restricted, e. g., by budget constraints. More generally, suppose there is a convex function $g : \mathbb{R}^m \to \mathbb{R}^k$, $k \in \mathbb{N}$ such that for any feasible solution $\boldsymbol{z}$ the condition $g(\boldsymbol{z}) \leq \mathbf{0}$ must be satisfied. The function g, for instance, can represent edge-specific budget constraints $\ell_e z_e \leq B_e$ for $e \in E$ and/or a global budget constraint $\sum_{e \in E} \ell_e z_e \leq B$. We arrive at the following budgeted continuous network design problem (bCNDP):

$$\min_{\boldsymbol{z} \geq 0} \min_{\boldsymbol{v} \in \mathcal{W}(\boldsymbol{z})} \sum_{e \in E} S_e(v_e/z_e)\, v_e \quad s.t. : g(\boldsymbol{z}) \leq \mathbf{0}. \tag{bCNDP}$$

Using existing results from the literature [6, 21], we can show a 4/3-approximation for affine latencies, and that there is there is no polynomial $(4/3 - \epsilon)$-approximation algorithm with $\epsilon > 0$ unless $\mathsf{P} = \mathsf{NP}$, see the full version of this paper. For proving the lower bound, we use edge-specific budget constraints and mimic a construction from Roughgarden [22]. It is an interesting open problem whether such a lower bound can also be achieved if we allow only a global budget constraint.

References

1 M. Abdulaal and L. J. LeBlanc. Continuous equilibrium network design models. *Transportation Res. Part B*, 13(B):19–32, 1979.

2 R. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice-Hall, Upper Saddle River, NJ, USA, 1993.

3 M. Beckmann, C. B. McGuire, and C. B. Winsten. *Studies in the Economics and Transportation*. Yale University Press, New Haven, CT, USA, 1956.

4 U. Bhaskar, K. Ligett, and L. J. Schulman. Network improvement for equilibrium routing. In *Proc. 17th Int. Conf. on Integer Programming and Combinatorial Optimization (IPCO)*, pages 138–149, 2014.

5 B. Colson, P. Marcotte, and G. Savard. An overview of bilevel optimization. *Annals of Oper. Res.*, 153(1):235–256, 2007.

6 J. Correa, A. Schulz, and N. Stier-Moses. Selfish routing in capacitated networks. *Math. Oper. Res.*, 29(4):961–976, 2004.

7 S. C. Dafermos. *Traffic assignment and resource allocation in transportation networks*. PhD thesis, John Hopkins University, Baltimore, MD, 1968.

8 S. C. Dafermos. Traffic equilibrium and variational inequalities. *Transportation Sci.*, 14:42–54, 1980.

9 G. B. Dantzig, R. P. Harvey, Z. F. Lansdowne, D. W. Robinson, and S. F. Maier. Formulating and solving the network design problem by decomposition. *Transportation Res. Part B*, 13(1):5–17, 1979.

10 T. L. Friesz. Transportation network equilibrium, design and aggregation: Key developments and research opportunities. *Transportation Res. Part A*, 17(B):411–426, 1985.

11 M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, Berlin, Germany, 2nd edition, 1993.

12 Y. A. Korilis, A. A. Lazar, and A. Orda. Architecting noncooperative networks. *IEEE J. Sel. Area Commun.*, 13(7):1241–1251, 1995.

13 Y. A. Korilis, A. A. Lazar, and A. Orda. Avoiding the Braess paradox in noncooperative networks. *J. Appl. Probab.*, 36(1):211–222, 1999.

14 L. Libman and A. Orda. The designer's perspective to atomic noncooperative networks. *IEEE/ACM Trans. Networking*, 7(6):875–884, 1999.

15 H. Lin, T. Roughgarden, É. Tardos, and A. Walkover. Stronger bounds on Braess's paradox and the maximum latency of selfish routing. *SIAM J. Comput.*, 25(4):1667–1686, 2011.

16 Z.-Q. Luo, J.-S. Pang, and D. Ralph. *Mathematical programs with equilibrium constraints*. Cambridge University Press, Cambridge, UK, 1996.

17 T. L. Magnanti and R. T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Sci.*, 18(1):1–55, 1984.

18 P. Marcotte. Network design problem with congestion effects: A case of bilevel programming. *Math. Progam.*, 34:142–162, 1986.

19 P. Marcotte and G. Marquis. Efficient implementation of heuristics for the continuous network design problem. *Annals of Oper. Res.*, 34:163–176, 1992.

20 L. Qiu, Y. R. Yang, Y. Zhang, and S. Shenker. On selfish routing in Internet-like environments. *IEEE/ACM Trans. Networking*, 14(4):725–738, 2006.

21 T. Roughgarden. The price of anarchy is independent of the network topology. *J. Comput. System Sci.*, 67:341–364, 2002.

22 T. Roughgarden. On the severity of Braess's paradox: Designing networks for selfish users is hard. *J. Comput. System Sci.*, 72(5):922–953, 2006.

23 T. Roughgarden and É. Tardos. How bad is selfish routing? *J. ACM*, 49(2):236–259, 2002.

24 M. J. Smith. The existence, uniqueness and stability of traffic equilibria. *Transportation Res.*, 13(B):295–304, 1979.

25 U. S. Bureau of Public Roads. *Traffic assignment manual*. U.S. Department of Commerce, Urban Planning Division, Washington, DC, 1964.

26 G. Valiant and T. Roughgarden. Braess's paradox in large random graphs. *Random Structures Algorithms*, 37(4):495–515, 2010.

27 H. Yang and M. G. H. Bell. Models and algorithms for road network design: a review and some new developments. *Transport Reviews*, 18(3):257–278, 1998.

Approximate Pure Nash Equilibria in Weighted Congestion Games*

Christoph Hansknecht[1], Max Klimm[1], and Alexander Skopalik[2]

1 **Department of Mathematics, Technische Universität Berlin**
Straße des 17. Juni 136, 10623 Berlin, Germany
{hansknecht,klimm}@math.tu-berlin.de

2 **Department of Computer Science, University of Paderborn**
Fürstenallee 11, 33102 Paderborn, Germany
skopalik@mail.uni-paderborn.de

Abstract

We study the existence of approximate pure Nash equilibria in weighted congestion games and develop techniques to obtain approximate potential functions that prove the existence of α-approximate pure Nash equilibria and the convergence of α-improvement steps. Specifically, we show how to obtain upper bounds for approximation factor α for a given class of cost functions. For example for concave cost functions the factor is at most $3/2$, for quadratic cost functions it is at most $4/3$, and for polynomial cost functions of maximal degree ℓ it is at at most $\ell + 1$. For games with two players we obtain tight bounds which are as small as for example 1.054 in the case of quadratic cost functions.

1998 ACM Subject Classification J.4 Computer applications – Social and Behavioral Sciences, C.2.2 Computer-communication networks – Network protocols

Keywords and phrases Congestion game, Pure Nash equilibrium, Approximate equilibrium, Existence, Potential function

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.242

1 Introduction

In many applications the state of a system depends on the behavior of individual participants that act selfishly in order to minimize their own private cost measured by individual objective functions. The framework of *non-cooperative games* has enveloped as the primary tool for the theoretical analysis of such systems. The central concept of game theory is that of a Nash equilibrium – a state in which no participant has an incentive to deviate to another strategy. While mixed Nash equilibria, i.e., Nash equilibria in randomized strategies, are guaranteed to exist under mild assumptions on the players' strategy spaces and the private cost functions (cf. Nash [17], Glicksberg [12]), they are often hard to interpret. As a consequence, attention is often restricted to pure Nash equilibria, i.e., Nash equilibria in deterministic strategies.

Rosenthal [19] introduced a rich class of games, called *congestion games* that models a wealth of strategic interactions and is guaranteed to have pure Nash equilibria. In a congestion game, we are given a finite set of players and a finite set of resources. A strategy

* This research was partially supported by the German Research Foundation (DFG) within the Collaborative Research Center "On-The-Fly Computing" (SFB 901). It was also partially supported by the EU within integrated project TEAM (contract no. 318621) and FET project MULTIPLEX (contract no. 317532).

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 242–257

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

of each player is to choose a subset of the resources out of a set of subsets of resources allowable to her. In each strategy profile, each player pays for all used resources where the cost of a resource is a function of the number of players using it. In most applications, the set of resources corresponds to the set of edges of a directed or undirected graph and cost functions are used to model latencies or travel times that typically increase with congestion. In that way, congestion games can be used to model traffic in road networks and Internet routing applications, where streams of packets choose a chain of servers from their origin to their destination.

Unfortunately, several mild generalizations of congestion games may lack pure Nash equilibria. In a *weighted* congestion game, each player is associated with a positive *demand* and the cost of each resource depends on the aggregated demand rather than the mere cardinality of the set of its users. It is well known that weighted congestion games may fail to have a pure Nash equilibrium; examples of such games have been given by Libman and Orda [15], Goemans et al. [13], and Fotakis et al. [11]. The games of Fotakis et al. and Goemans et al. feature two players with demands one and two, respectively. While Fotakis et al. specify for each resource explicitly the cost for all possible aggregated demands, Goemans et al. use only polynomial cost functions with non-negative coefficients and maximal degree two. Full enumeration of all strategy profiles shows that no pure Nash equilibrium exists. Quite strikingly, however, both examples admit deterministic states that are *almost* stable, i.e., there are strategy profiles from which both players may only improve by a small factor. In the game of Fotakis et al. (with no structure on the cost functions), this factor is as little as $12/11$, in the game of Goemans et al., this factor is $61/60$. Put differently, if there is some friction in the system that prevents players from making deviations that improve their private costs only very little, then stable states exist.

Such an approximate stability is formally captured by the concept of an α-approximate pure Nash equilibrium, a state from which no player can improve her private cost by a factor of $\alpha \geq 1$. Besides mere existence, approximate equilibria are an appealing alternative solution concept from a computational point of view. While the computation of exact pure Nash equilibria in congestion games is PLS-complete, there has been some recent progress towards polynomial time algorithms to compute approximate equilibria in congestion games. Specifically, Caragiannis et al. [4] show how to compute a $2 + \epsilon$-approximate pure Nash equilibrium in congestion games with affine latencies. Subsequent work generalizes this approach to a polynomial algorithm for approximate pure Nash equilibria with constant approximation factor for weighted congestion games with polynomial latency functions [5]. They also show that weighted congestion games with polynomial cost functions with maximal degree ℓ have a $\ell!$-approximate pure Nash equilibrium.

Still, approximate pure Nash equilibria are only a reasonable concept if the approximation factor is sufficiently close to one. This motivates the main question of this paper: Given a set of cost functions, what is the minimal approximation factor α that one can allow in order to guarantee the existence of an α-approximate pure Nash equilibrium in all weighted congestion games?

1.1 Our Contribution

The main tool to answer this question are *approximate potential functions*. An α-approximate potential is a map from the space of all strategy profiles to the real numbers that has the property that it decreases if a player decreases his cost by a factor greater than α. Note that, unlike an exact potential function, for improvement steps of smaller relative size, an approximate potential function may actually increase. The existence of α-approximate

Table 1 Approximation factors of approximate pure Nash equilibria in weighted congestion games for different sets of cost functions; results for two-player games are tight. Comparison with previous results [5] shows improvements by a factor exponential in ℓ.

	Our results		Previous work [5]
Functions	≥ 3 players	2 players	≥ 3 players
concave	$\leq 3/2$		
polynomials of degree 2	$\leq 4/3$	≈ 1.054	≤ 2
— " — 3	≤ 1.785	≈ 1.074	≤ 6
— " — 4	≤ 2.326	≈ 1.153	≤ 24
— " — ℓ	$\leq \ell + 1$		$\leq \ell!$

potential functions immediately implies the existence of α-approximate equilibria and the convergence of α-improvement steps. We present two methods to obtain an α-approximate potential function and identify upper bounds for the value of α for a given class of cost functions.

Our technique yields small approximation factors for specific classes of cost functions summarized in Table 1. For concave cost functions we establish the existence of $\frac{3}{2}$-approximate equilibria. For quadratic cost functions the factor is at most $\frac{4}{3}$ in games with an arbitrary number of players. More surprisingly, in games with two players, we obtain a tight bound of about 1.054 using numerical methods. This shows that the factor of $61/60 \approx 1.017$ achieved by the two-player game of Goemans et al. is not so far from the worst-case bound for arbitrary two-player games with quadratic costs.

For polynomial cost functions of maximal degree ℓ, we obtain an upper bound of $\ell + 1$ which is a drastic improvement of the previously known bound of $\ell!$.

Our improved bounds on the minimal approximation factors for pure Nash equilibria may be used to design routing protocols with convergent behavior. While it is known that routing with distance vector computation causes flapping, our results suggest that for routers with quadratic latencies, routes should not be updated as long as the new route does not improve latency by a factor of at least 4/3.

1.2 Further Related Work

Rosenthal [19] proved that every congestion game has a pure Nash equilibrium using an elegant *potential function* argument. A potential function assigns a real value to each strategy profile such that for two profiles which differ only in the strategy choice of one player the cost difference for that player equals the difference of the two potential function values. This property implies that any sequence of improvement steps by single players converges to a pure Nash equilibrium. However, such a sequence might take exponentially many steps and computing a pure Nash equilibrium is a computationally hard task as it is PLS-hard [1, 3, 10].

In contrast to the original class of unweighted congestion games studied by Rosenthal, many natural generalizations do neither have a potential function nor a pure Nash equilibrium, in general. Milchtaich [16] introduced weighted congestion games and congestion games with player-specific cost functions. He restricts himself to the singleton case, where each strategy of each player contains a single resource only and showed that games with player-specific cost always have a pure Nash equilibrium if players are unweighted.

Fotakis et al. [11] study congestion games with weighted players and arbitrary strategy spaces. They show that pure Nash equilibria need not exist in general. This has been observed independently by Goemans et al. [13] and Libman and Orda [15]. The computation problem to decide whether a pure Nash equilibrium exists is NP-hard [9]. For the special case of only affine [11] or exponential [18] cost functions the existence of a pure Nash equilibrium is guaranteed. These results are complemented by the characterization of Harks and Klimm [14] who prove that these are the only cost functions that guarantee the existence. These results are entirely independent of the underlying structure of the games. In contrast, Ackermann et al. [2] consider weighted congestion games with arbitrary cost functions but restrictions on the combinatorial structure of strategy spaces. They prove existence of pure Nash equilibria for weighted congestion games that have the *matroid property*, i.e. the property that the set of possible strategies for different players forms a matroid.

In light of the negative results regarding existence and complexity of pure Nash equilibria, attention turned towards approximate equilibria. For symmetric and unweighted congestion games, Chien and Sinclair [7] showed fast convergence to approximate equilibria under some mild assumption on the cost functions. However, Skopalik and Vöcking [20] proved that in the asymmetric case it is PLS-hard to compute an α-approximate equilibrium for any polynomial time computable α. As it turns out, this seemingly devastating result relies on the use of cost functions that allow negative coefficients. Indeed Caragiannis et al. [4] presented a polynomial time algorithm to compute approximate equilibria in unweighted congestion games with linear and polynomial cost functions without negative coefficients. In subsequent work [5], the same authors study the existence and complexity of approximate equilibria in weighted congestion games. In particular they show that a game with polynomial delay functions of degree at most d, a $d!$-approximate equilibrium always exists. They introduce a new class of games called Ψ-games. A weighted congestion game is approximated by a corresponding Ψ-game. That is, the cost a player in a weighted congestion game is approximates by her cost in the corresponding Ψ-game up factor of $d!$. These Ψ-games are potential games which immediately proves the existence of $d!$-approximate equilibria. Using a similar algorithm as in [4] one can compute $d^{d+o(1)}$-approximate equilibria in games with polynomial delay functions and $\frac{3+\sqrt{5}}{2}$-approximate equilibria in the case of linear weighted congestion games.

Chen and Roughgarden [6] studied approximate equilibria in network design games with weighted players and showed existence of approximate equilibria using approximate potential functions. They were also used by Christodoulou et al. [8] in order to derive tight bounds on the price of anarchy and price of stability of approximate pure Nash equilibria in unweighted congestion games.

2 Preliminaries

We consider finite strategic games $G = (N, \mathbf{S}, \pi)$, where $N = \{1, \dots, n\}$ is the non-empty and finite set of *players*, S_i is the set of *strategies* available to player i, $\mathbf{S} = S_1 \times \ldots \times S_n$ is the finite and non-empty set of *strategy profiles*, $\pi_i : \mathbf{S} \to \mathbb{R}^n$ is the *private cost function* player i strives to minimize, and $\pi : \mathbf{S} \to \mathbb{R}^n$, $\mathbf{s} \mapsto \pi_1(\mathbf{s}) \times \cdots \times \pi_n(\mathbf{s})$ is the *combined private cost function*.

Vectors of sets and vectors of real numbers are denoted with bold face. We use standard game theory notation, i.e., for a player i and a strategy profile $\mathbf{s}$, we write $\mathbf{s} = (s_i, \mathbf{s}_{-i})$ meaning that $s_i \in S_i$ and $\mathbf{s}_{-i} \in \mathbf{S}_{-i} = S_1 \times \cdots \times S_{i-1} \times S_{i+1} \times \cdots \times S_n$. For $\alpha \geq 1$, a strategy profile $\mathbf{s}$ is an α-approximate Nash equilibrium, if $\pi_i(\mathbf{s}) \leq \alpha \cdot \pi_i(t_i, \mathbf{s}_{-i})$ for all $i \in N$ and $t_i \in S_i$. For $\alpha = 1$, we call $\mathbf{s}$ a Nash equilibrium rather than a 1-approximate Nash equilibrium.

In a *weighted congestion game*, we are given a demand vector $\mathbf{d} = (d_1, \ldots, d_n) \in \mathbb{R}^n_{>0}$ specifying a positive demand d_i for each player i. The set of strategies of player i is a non-empty set $S_i \subseteq 2^R$ of subsets of a given set of resources R. Given a strategy profile $\mathbf{s} \in \mathbf{S}$, we denote by $N_r(\mathbf{s}) = \{i \in N : r \in s_i\}$ the set of players that use r in $\mathbf{s}$. The aggregated demand for resource r is denoted by $|\mathbf{d}_{N_r(\mathbf{s})}|$. Each resource is endowed with a cost function $c_r : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ that maps the aggregated demand $|\mathbf{d}_{N_r(\mathbf{s})}|$ to a cost value $c_r(|\mathbf{d}_{N_r(\mathbf{s})}|)$. The private cost of player i is then defined as $\pi_i(\mathbf{s}) = \sum_{r \in s_i} c_r(|\mathbf{d}_{N_r(\mathbf{s})}|)$.

A weighted congestion game in which all players have unit demand $d_i = 1$ is called an *unweighted congestion game*. Rosenthal proved that every unweighted congestion game has a pure Nash equilibrium by giving a potential function. For $\alpha \geq 1$, an α-approximate *potential function* is a map $P : \mathbf{S} \mapsto \mathbb{R}$ with the property that $P(t_i, \mathbf{s}_{-i}) < P(\mathbf{s})$ whenever $\alpha\pi_i(t_i, \mathbf{s}_{-i}) < \pi_i(\mathbf{s})$ for some $i \in N$, $t_i \in S_i$, and $\mathbf{s} \in \mathbf{S}$. In case $\alpha = 1$, we call P a potential function. Rosenthal [19] showed that $P : \mathbf{S} \to \mathbb{R}$ defined as $P(\mathbf{s}) = \sum_{r \in R} \sum_{x=0}^{|\mathbf{d}_{N_r(s)}|} c_r(x)$ is a potential function for unweighted congestion games.

3 Existence of an Approximate Potential Function

In this section, we show that weighted congestion games admit approximate potential functions with low approximation factor. Roughly speaking, we obtain an approximate potential as follows. For each resource, we choose an appropriate ordering of the players. Then, for each resource separately, we compute a *discrete integral*: We sum up the resource cost after introducing the first player multiplied with the first players' demand, the resource cost after introducing the first two players multiplied with the second players' demand, and so on. When the demands of all players go to zero while keeping the total demand of all players constant this value approaches the integral from zero to the total demand of the cost function, hence the name.

To abstract from the underlying set of players, we define the discrete integral of a function with respect to an arbitrary vector $\mathbf{v} \in \mathbb{R}^n_{>0}$.

▶ **Definition 1** (Discrete integral)**.** Let $\mathbf{v} \in \mathbb{R}^n_{>0}$ and let $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$. The $\mathbf{v}$-discrete integral is defined as $I_f(\mathbf{v}) = \sum_{i=1}^{n} v_i f(\sum_{j=1}^{i} v_j)$.

To obtain provably low approximation factors, we are interested in permutations of the vector $\mathbf{v}$ that minimize or maximize the value of the discrete integral among all permutations of the vector. For a vector $\mathbf{v} \in \mathbb{R}^n_{>0}$, and a function f we let denote $\hat{\sigma}_f(\mathbf{v})$ the vector in $\mathbb{R}^n_{>0}$, that contains the entries of $\mathbf{v} = (v_1, \ldots, v_n)$ in an order that minimizes the resulting discrete integral, i.e., $I_f(\hat{\sigma}_f(\mathbf{v})) \leq I_f((v_{p(1)}, \ldots, v_{p(n)}))$ for all permutations $p \in \Pi$. With a slight abuse of terminology, we call $\hat{\sigma}_f(\mathbf{v})$ a *permutation* of $\mathbf{v}$. If several permutations of $\mathbf{v}$ achieve the same value for the discrete integral, we assume that ties are broken according to an arbitrary, but fixed tie-breaking rule. Similarly, we denote by $\check{\sigma}_f(\mathbf{v})$ the permutation of $\mathbf{v}$ that maximizes the resulting value of the discrete integral, i.e., $I_f(\check{\sigma}_f(\mathbf{v})) \geq I_f((v_{p(1)}, \ldots, v_{p(n)}))$ for all permutations $p \in \Pi$. We sometimes call $\hat{\sigma}_f(\mathbf{v})$ and $\check{\sigma}_f(\mathbf{v})$ simply *minimizing* and *maximizing* permutations of $\mathbf{v}$ when the underlying function f is clear from the context.

Given a minimizing or maximizing permutation $\hat{\sigma}_f(\mathbf{v})$ or $\check{\sigma}_f(\mathbf{v})$ of a vector $\mathbf{v} = (v_1, \ldots, v_n)$, we are interested in the (relative) error in the minimization or maximization of the discrete integral when moving a given entry v_i to the end of the permutation. Formally, let $\mathbf{v} \in \mathbb{R}^n_{>0}$ and let $p \in \Pi$ be such that $\hat{\sigma}_f(\mathbf{v}) = (v_{p(1)}, \ldots, v_{p(n)})$. Further, let $i, j \in \{1, \ldots, n\}$ be such that $v_i = v_{p(j)}$. Then, we denote by $\hat{\sigma}^i_f(\mathbf{v})$ the vector that arises from $\hat{\sigma}_f(\mathbf{v})$ by moving entry $v_{p(j)}$ to the end of the vector, i.e., $\hat{\sigma}^i_f(\mathbf{v}) = (v_{p(1)}, \ldots, v_{p(j-1)}, v_{p(j+1)}, \ldots, v_{p(n)}, v_{p(j)})$.

Analogously, $\check{\sigma}^i_f(\mathbf{v})$ is the vector that arises from the maximizing permutation $\check{\sigma}_f(\mathbf{v})$ by moving the entry corresponding to v_i to the end of the vector.

The approximation factors for approximate equilibria that we are going to obtain depend on the following relative errors in the minimization respectively maximization of the discrete integral after moving entry v_i to the end of the vector.

▶ **Definition 2.** For a function $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, define

$$\hat{\mu}(f) = \sup_{\mathbf{v}\in\mathbb{R}^n_{>0}} \max_{i\in\{1,\dots,n\}} \frac{I_f(\hat{\sigma}^i_f(\mathbf{v})) - I_f(\hat{\sigma}_f(\mathbf{v}))}{v_i f(|\mathbf{v}|)},$$
$$\check{\mu}(f) = \sup_{\mathbf{v}\in\mathbb{R}^n_{>0}} \max_{i\in\{1,\dots,n\}} \frac{I_f(\check{\sigma}_f(\mathbf{v}) - I_f(\check{\sigma}^i_f(\mathbf{v}))}{v_i f(|\mathbf{v}|)}.$$

For a set $\mathcal{C}$ of functions, we set $\check{\mu}(\mathcal{C}) = \sup_{f\in\mathcal{C}} \check{\mu}(f)$ and $\hat{\mu}(\mathcal{C}) = \sup_{f\in\mathcal{C}} \hat{\mu}(f)$, respectively.

The following theorem relates the existence of an α-approximate pure Nash equilibrium in a weighted congestion game with cost functions in $\mathcal{C}$ with the value $\check{\mu}(\mathcal{C})$.

▶ **Theorem 3.** *Every weighted congestion game with cost functions in $\mathcal{C}$ has an α-approximate Nash equilibrium with $\alpha = 1 + \check{\mu}(\mathcal{C})$.*

Proof. To prove the result, we show that the function $P : \mathbf{S} \to \mathbb{R}$ defined as $P(\mathbf{s}) = \sum_{r\in R} I_{c_r}(\check{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))$ is an α-approximate potential function.

Consider an arbitrary strategy profile $\mathbf{s} \in S$ and let $i \in N$ and $\mathbf{t} \in S$ be such that $\mathbf{t} = (t_i, \mathbf{s}_{-i})$ for some $t_i \in S_i$ with $\alpha \cdot \pi_i(t_i, \mathbf{s}_{-i}) < \pi_i(\mathbf{s})$. We calculate

$$\begin{aligned} P(\mathbf{t}) - P(\mathbf{s}) &= \sum_{r\in R}\Big(I_{c_r}(\check{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\check{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\Big) \\ &= \sum_{r\in R}\Big(I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\Big) && \text{(1a)} \\ &\quad + \sum_{r\in R}\Big(I_{c_r}(\check{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})}))\Big) && \text{(1b)} \\ &\quad + \sum_{r\in R}\Big(I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))) - I_{c_r}(\check{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\Big). && \text{(1c)} \end{aligned}$$

We proceed to bound the expressions (1a), (1b), and (1c) separately, starting with (1a). Clearly, for all resources $r \in R \setminus (s_i \triangle t_i)$, the discrete integrals in (1a) cancel out, so we only have to consider the resources in $s_i \setminus t_i$ and $t_i \setminus s_i$. By definition, for all resources $r \in s_i \setminus t_i$, the demand d_i appears last in $\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})})$ but not at all in $\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})$ and, thus, $I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})})) = -d_i c_r(|\mathbf{d}_{N_r(\mathbf{s})}|)$. Analogously, for a resource $r \in t_i \setminus s_i$, the demand d_i appears last in $\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})$, but not at all in $\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})})$ and we obtain $I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})})) = d_i c_r(|\mathbf{d}_{N_r(\mathbf{t})}|)$. Thus, we may rewrite (1a) as

$$\sum_{r\in R}\Big(I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\Big) = d_i\big(\pi_i(\mathbf{t}) - \pi_i(\mathbf{s})\big).$$

Next, consider the expression (1b). Using the definition of $\check{\mu}(f)$, we bound (1b) from above by

$$\sum_{r\in R}\Big(I_{c_r}(\check{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{t})}))) - I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})}))\Big) \leq \sum_{r\in t_i} \check{\mu}_{c_r} \cdot d_i c_r(|\mathbf{d}_{N_r(\mathbf{t})}|) \leq \check{\mu}(\mathcal{C}) \cdot d_i\, \pi_i(\mathbf{t}).$$

Finally, for expression (1c), by definition of $\check{\sigma}_{c_r}$ we have $I_{c_r}\big(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\big) < I_{c_r}\big(\check{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{s})})\big)$ for all $r \in R$ and, thus,

$$\sum_{r\in R}\Big(I_{c_r}(\check{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))) - I_{c_r}(\check{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\Big) < 0. \tag{2a}$$

Plugging everything together, we obtain

$$P(\mathbf{t}) - P(\mathbf{s}) \leq d_i\big(\pi_i(\mathbf{t}) - \pi_i(\mathbf{s})\big) + \check{\mu}(\mathcal{C}) \cdot d_i\, \pi_i(\mathbf{t}) = d_i\big(\alpha\pi_i(\mathbf{t}) - \pi_i(\mathbf{s})\big) < 0.$$

We conclude that P is an α-approximate potential. As the set of strategy profiles is finite, P attains its minimum on $\mathbf{S}$ which is an α-approximate Nash equilibrium. ◀

We obtain a similar bound by choosing for each resource an order of the players' demands that *minimizes* the resulting discrete integral.

▶ **Theorem 4.** *Every weighted congestion game with cost functions in $\mathcal{C}$ and $\hat{\mu}(\mathcal{C}) < 1$ has an α-approximate Nash equilibrium with $\alpha = \big(1 - \hat{\mu}(\mathcal{C})\big)^{-1}$.*

Proof. Let $\alpha = \frac{1}{1-\hat{\mu}(\mathcal{C}))}$. We consider the function $P : \mathbf{S} \to \mathbb{R}$ defined as $P(\mathbf{s}) = \sum_{r\in R} I_{c_r}(\hat{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))$ and show that it is an α-approximate potential function. To this end, let us again consider an arbitrary strategy profile $\mathbf{s} \in S$ and let $i \in N$ and $\mathbf{t} \in S$ be such that $\mathbf{t} = (t_i, \mathbf{s}_{-i})$ for some $t_i \in S_i$ with $\alpha \cdot \pi_i(t_i, \mathbf{s}_{-i}) \leq \pi_i(\mathbf{s})$. We calculate

$$\begin{aligned}
P(\mathbf{s}) - P(\mathbf{t}) &= \sum_{r\in R}\Big(I_{c_r}(\hat{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\hat{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\Big) \\
&= \sum_{r\in R}\Big(I_{c_r}(\hat{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\hat{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\Big) && \text{(3a)} \\
&\quad + \sum_{r\in R}\Big(I_{c_r}(\hat{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{t})})) - I_{c_r}(\hat{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{t})}))\Big) && \text{(3b)} \\
&\quad + \sum_{r\in R}\Big(I_{c_r}(\hat{\sigma}^i_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))) - I_{c_r}(\hat{\sigma}_{c_r}(\mathbf{d}_{N_r(\mathbf{s})}))\Big). && \text{(3c)}
\end{aligned}$$

We proceed along the same lines as in the proof of Theorem 3. Note that this time the expression (2a) does not exceed 0 as we chose an order that minimizes the discrete integral. Using the definition of $\hat{\mu}(\mathcal{C})$, we bound the expression in(3c) from above by $\hat{\mu}(\mathcal{C}) \cdot d_i\, \pi_i(\mathbf{s})$. Plugging everything together, we obtain

$$\begin{aligned}
P(\mathbf{t}) - P(\mathbf{s}) &\leq d_i\big(\pi_i(\mathbf{t}) - \pi_i(\mathbf{s})\big) + \hat{\mu}(\mathcal{C}) \cdot d_i\, \pi_i(\mathbf{s}) \\
&= d_i\big(\pi_i(\mathbf{t}) - (1 - \hat{\mu}(\mathcal{C}))\pi_i(\mathbf{s})\big) \\
&= \frac{d_i}{\alpha}\big(\alpha\pi_i(\mathbf{t}) - \pi_i(\mathbf{s})\big) < 0.
\end{aligned}$$

We conclude that P is an α-approximate potential. ◀

4 Bounding the Approximation Factor

To obtain explicit numerical bounds on the approximation factor of the approximate Nash equilibria, we want to compute $\check{\mu}(\mathcal{C})$ and $\hat{\mu}(\mathcal{C})$ for interesting sets $\mathcal{C}$ of cost functions. As a first result in this direction, we show that for a convex function f and a vector $\mathbf{v} \in \mathbb{R}^n_{>0}$, the discrete integral is maximized when $\mathbf{v}$ is sorted in non-decreasing order.

Formally, for a vector $\mathbf{v} \in \mathbb{R}^n_{>0}$, let us denote by $\overleftarrow{\sigma}(\mathbf{v}) = (v_{p(1)}, \dots, v_{p(n)})$, $p \in \Pi$ the permutation of $\mathbf{v} = (v_1, \dots, v_n)$ that contains the entries of $\mathbf{v}$ in non-increasing order, i.e, $v_{p(1)} \geq v_{p(2)} \geq \cdots \geq v_{p(n)}$. Equivalently, we denote by $\vec{\mathbf{v}}$ the permutation of $\mathbf{v}$ that contains the entries of $\mathbf{v}$ in non-decreasing order.

▶ **Lemma 5** (Sorting Lemma). *For all $\mathbf{v} \in \mathbb{R}^n_{>0}$ and $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ the following statements hold:*

1. *If f is convex, then $I_f(\hat{\sigma}_f(\mathbf{v})) = I_f(\overleftarrow{\sigma}(\mathbf{v}))$ and $I_f(\check{\sigma}_f(\mathbf{v})) = I_f(\vec{\sigma}(\mathbf{v}))$.*
2. *If f is concave, then $I_f(\hat{\sigma}_f(\mathbf{v})) = I_f(\vec{\sigma}(\mathbf{v}))$ and $I_f(\check{\sigma}(\mathbf{v})) = I_f(\overleftarrow{\sigma}(\mathbf{v}))$.*

Proof. We start proving *1.* Let $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ be a convex function and let $\mathbf{v} \in \mathbb{R}^n_{>0}$ be arbitrary. It suffices to prove $I_f(\overleftarrow{\sigma}(\mathbf{v})) \leq I_f(\hat{\sigma}_f(\mathbf{v}))$. Let $\mathbf{w} = (w_1, \dots, w_n) = \hat{\sigma}_f(\mathbf{v})$. If $\mathbf{w} = (w_1, w_2, \dots, w_n)$ is already non-increasing, there is nothing left to show. Otherwise, there is $i \in \{1, \dots, n-1\}$ with $w_i < w_{i+1}$. Consider the vector $\mathbf{w}'$ that is obtained from $\mathbf{w}$ by swapping entries w_i and w_{i+1}, i.e. $\mathbf{w}' = (w'_1, w'_2, \dots, w'_n)$ where

$$w'_j = \begin{cases} w_j, & \text{if } j \notin \{i, i+1\}, \\ w_{i+1}, & \text{if } j = i, \\ w_i, & \text{if } j = i+1. \end{cases}$$

We claim that $I_f(\mathbf{w}') \leq I_f(\mathbf{w})$. To see this claim, let us denote by $x = \sum_{j=1}^{i-1} w_j$ the sum of the entries of $\mathbf{w}$ with index smaller than i. As f is convex, we have $f(\lambda a + (1-\lambda)b) \leq \lambda f(a) + (1-\lambda) f(b)$ for all $\lambda \in (0,1)$, $a, b \in \mathbb{R}_{\geq 0}$. For $\lambda = w_i / w_{i+1}$, $a = x + w_i$, and $b = x + w_i + w_{i+1}$, we obtain in particular

$$f(x + w_{i+1}) \leq \frac{w_i}{w_{i+1}} f(x + w_i) + \Big(1 - \frac{w_i}{w_{i+1}}\Big) f(x + w_i + w_{i+1}) \tag{4a}$$

which implies

$$w_{i+1} f(x + w_{i+1}) + v_i f(x + w_i + w_{i+1}) \leq w_i f(x + w_i) + w_{i+1} f(x + w_i + w_{i+1}). \tag{4b}$$

We derive that $I_f(\mathbf{w}') \leq I_f(\mathbf{w})$. By iteratively switching entries w_j, and w_{j+1} with $w_j < w_{j+1}$ we transform $\mathbf{w}$ into $\overleftarrow{\sigma}(\mathbf{v})$ without increasing the value of the discrete integral. This proves $I_f(\overleftarrow{\sigma}(\mathbf{v})) \leq I_f(\mathbf{w})$.

For concave functions, all inequality signs in (4a) and (4b) reverse and *2.* is obtained along the same lines. ◀

We proceed to provide a series of useful lemmas that help to identify the structure of the worst case vectors $\mathbf{v}$ that get arbitrarily close to the suprema

$$\hat{\mu}(f) = \sup_{\mathbf{v} \in \mathbb{R}^n_{>0}} \max_{i \in \{1,\dots,n\}} \big(I_f(\hat{\sigma}^i_f(\mathbf{v})) - I_f(\hat{\sigma}_f(\mathbf{v}))\big) \big/ \big(v_i f(|\mathbf{v}|)\big), \qquad \text{and}$$
$$\check{\mu}(f) = \sup_{\mathbf{v} \in \mathbb{R}^n_{>0}} \max_{i \in \{1,\dots,n\}} \big(I_f(\check{\sigma}_f(\mathbf{v}) - I_f(\check{\sigma}^i_f(\mathbf{v}))\big) \big/ \big(v_i f(|\mathbf{v}|)\big),$$

respectively.

Roughly speaking, in the next lemma, we show that for a convex function f and a vector $\mathbf{v} = (v_1, \dots, v_i, \dots, v_j, v_{j+1}, \dots, v_n)$ the value $I_f(\check{\sigma}_f(\mathbf{v}) - I_f(\check{\sigma}^i_f(\mathbf{v}))$ cannot decrease when merging the two entries v_j and v_{j+1} to a single entry $v_j + v_{j+1}$. For a concave function, the same statement holds for $I_f(\hat{\sigma}^i_f(\mathbf{v})) - I_f(\hat{\sigma}_f(\mathbf{v}))$. To handle the convex and the concave case simultaneously, the statement of the lemma is slightly more general and holds for arbitrary vectors $\mathbf{v}$ rather than only those that are minimizing or maximizing.

▶ **Lemma 6** (Union Lemma). *Let $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, and let $\mathbf{v}, \mathbf{v}^i \in \mathbb{R}^n_{>0}$, $\tilde{\mathbf{v}}, \tilde{\mathbf{v}}^i \in \mathbb{R}^{n-1}_{>0}$ be such that*

$$\mathbf{v} = (v_1, \ldots, v_i, \ldots, v_j, v_{j+1} \ldots, v_n), \qquad \mathbf{v}^i = (v_1, \ldots, v_j, v_{j+1}, \ldots, v_n, v_i),$$
$$\tilde{\mathbf{v}} = (v_1, \ldots, v_i, \ldots, v_j + v_{j+1}, \ldots, v_n), \qquad \tilde{\mathbf{v}}^i = (v_1, \ldots, v_j + v_{j+1}, \ldots, v_n, v_i)$$

i.e., $\tilde{\mathbf{v}}$ is obtained from $\mathbf{v}$ by joining entry v_j and v_{j+1}, $j > i$ into one entry $v_j + v_{j+1}$ and $\mathbf{v}^i$ and $\tilde{\mathbf{v}}^i$ are obtained from $\mathbf{v}$ and $\tilde{\mathbf{v}}$, respectively, by moving entry v_i to the end of the vector. Then, $I_f(\mathbf{v}^i) - I_f(\mathbf{v}) - \big(I_f(\tilde{\mathbf{v}}^i) - I_f(\tilde{\mathbf{v}})\big)$ is non-negative, if f is convex and non-positive, if f is concave.

Proof. Let $x = \sum_{k=1}^{j-1} v_j$ denote the sum of the entries that appear before v_j in $\mathbf{v}$ and $\tilde{\mathbf{v}}$. We obtain

$$\begin{aligned}
&I_f(\mathbf{v}^i) - I_f(\tilde{\mathbf{v}}^i) - \big(I_f(\mathbf{v}) - I_f(\tilde{\mathbf{v}})\big) \\
&= v_j f(x - v_i + v_j) + v_{j+1}(x - v_i + v_j + v_{j+1}) \\
&\quad - (v_j + v_{j+1}) f(x - v_i + v_j + v_{j+1}) \\
&\quad - \big(v_j f(x + v_j) + v_{j+1} f(x + v_j + v_{j+1}) - (v_j + v_{j+1}) f(x + v_j + v_{j+1})\big) \\
&= v_j \Big(f(x - v_i + v_j) - f(x - v_i + v_j + v_{j+1}) - \big(f(x + v_j) - f(x + v_j + v_{j+1})\big)\Big),
\end{aligned}$$

which is clearly non-positive for convex f and non-negative for concave f. ◀

By reversing the roles of $\mathbf{v}$ and $\tilde{\mathbf{v}}$ in the Union Lemma (Lemma 6) we obtain the following Split Lemma as a direct corollary of the Union Lemma.

▶ **Lemma 7** (Split Lemma). *Let $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, and let $\mathbf{v}, \mathbf{v}^i \in \mathbb{R}^n_{>0}$, $\tilde{\mathbf{v}}, \tilde{\mathbf{v}}^i \in \mathbb{R}^{n+1}_{>0}$ be such that*

$$\mathbf{v} = (v_1, \ldots, v_i, \ldots, v_j, \ldots, v_n), \qquad \mathbf{v}^i = (v_1, \ldots, v_j, \ldots, v_n, v_i),$$
$$\tilde{\mathbf{v}} = (v_1, \ldots, v_i, \ldots, v_j/2, v_j/2, \ldots, v_n), \qquad \tilde{\mathbf{v}}^i = (v_1, \ldots, v_j/2, v_j/2 \ldots, v_n, v_i)$$

i.e., $\tilde{\mathbf{v}}$ is obtained from $\mathbf{v}$ by splitting entry v_j, $j > i$ into two entries of size $v_j/2$ and $\mathbf{v}^i$ and $\tilde{\mathbf{v}}^i$ are obtained from $\mathbf{v}$ and $\tilde{\mathbf{v}}$, respectively, by moving entry d_i to the end of the vector. Then, $I_f(\mathbf{v}^i) - I_f(\mathbf{v}) - \big(I_f(\tilde{\mathbf{v}}^i) - I_f(\tilde{\mathbf{v}})\big)$ is non-positive, if f is convex and non-negative, if f is concave.

We proceed to use the Union Lemma (Lemma 6) to devise a closed form expression for $\breve{\mu}(f)$ for convex f respectively $\hat{\mu}(f)$ for concave f.

▶ **Corollary 8.** *For a function $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, let*

$$B_f(x, y, z) = \frac{x f(x + z) + y f(x + y + z) - y f(y + z) - x f(x + y + z)}{x f(x + y + z)}.$$

If f is convex, then $\breve{\mu}(f) = \sup_{x,y,z \in \mathbb{R}_{>0}, y \geq x} B_f(x, y, z)$, and if f is concave, then $\hat{\mu}(f) = \sup_{x,y,z \in \mathbb{R}_{>0}, y \geq x} -B_f(x, y, z)$.

Proof. We start to prove $\breve{\mu}(f) =_{x,y,z \in \mathbb{R}_{>0}, y \geq x} B(f)$, if f is convex. First, recall that $\breve{\mu}(f)$ is defined as

$$\breve{\mu}(f) = \sup_{\mathbf{v} \in \mathbb{R}^n_{>0}} \max_{i \in \{1, \ldots, n\}} \frac{I_f(\breve{\sigma}_f(\mathbf{v})) - I_f(\breve{\sigma}^i_f(\mathbf{v}))}{v_i f(|\mathbf{v}|)}.$$

By Lemma 5, it is without loss of generality to assume that $\check{\sigma}_f(\mathbf{v})$ contains the entries of $\mathbf{v}$ in non-decreasing order. Using this observation, we derive that there are sequences $(n_k)_{n\in\mathbb{N}}$, $n_k \in \mathbb{N}$ for all $k \in \mathbb{N}$, $(i_k)_{k\in\mathbb{N}}$, $i_k \in \{1,\dots,n_k\}$ for all $k \in \mathbb{N}$ and a sequence of non-decreasing vectors $(\mathbf{v}_k)_{k\in\mathbb{N}}$, $\mathbf{v}_k \in \mathbb{R}^{n_k}_{>0}$ such that

$$\lim_{k\to\infty} \frac{I_f(\mathbf{v}_k) - I_f(\mathbf{v}_k^{i_k})}{v_i f(|\mathbf{v}_k|)},$$

where we denote by $\mathbf{v}_k^{i_k}$ the vector that arises from $\mathbf{v}_k$ by moving the i_kth entry to the end of the vector. For ease of exposition, let us set $n = n_k$, $i = i_k$, $\mathbf{v} = (v_1,\dots,v_n) = \mathbf{v}_k$, $\mathbf{v}^i = (v_1,\dots,v_{i-1},v_{i+1},\dots,v_n,v_i) = v_k^{i_k}$ for some fixed $k \in \mathbb{N}$. Let $z = \sum_{j=1}^{i-1} v_j$ denote the sum of the entries in $\mathbf{v}$ before entry v_i, let $x = v_i$, and let $y = \sum_{j=i+1}^{n}$. Using the Union Lemma (Lemma 6), we derive that $I_f(\mathbf{v}) - I_f(\mathbf{v}^i)$ is maximal if $i+1 = n$, i.e., there is only one entry $v_{i+1} = y$ that appears after v_i in $\mathbf{v}$. We then obtain

$$\frac{I_f(\mathbf{v}) - I_f(\mathbf{v}^i)}{v_i f(|\mathbf{v}|)} = \frac{xf(x+z) + yf(x+y+z) - yf(y+z) - xf(x+y+z)}{xf(x+y+z)},$$

which proves the result.

For concave f, recall that $\hat{\mu}(f)$ is defined as

$$\check{\mu}(f) = \sup_{\mathbf{v}\in\mathbb{R}^n_{>0}} \max_{i\in\{1,\dots,n\}} \frac{I_f(\hat{\sigma}_f^i(\mathbf{v})) - I_f(\hat{\sigma}_f(\mathbf{v}))}{v_i f(|\mathbf{v}|)},$$

and that, by Lemma 5, it is without loss of generality to assume that $\hat{\sigma}_f(\mathbf{v})$ contains the entries of $\mathbf{v}$ in non-decreasing order. Analogously to the convex case, this implies the existence of sequences $(n_k)_{n\in\mathbb{N}}$, $n_k \in \mathbb{N}$ for all $k \in \mathbb{N}$, $(i_k)_{k\in\mathbb{N}}$, $i_k \in \{1,\dots,n_k\}$ for all $k \in \mathbb{N}$ and a sequence of non-decreasing vectors $(\mathbf{v}_k)_{k\in\mathbb{N}}$, $\mathbf{v}_k \in \mathbb{R}^{n_k}_{>0}$ such that

$$\lim_{k\to\infty} \frac{I_f(\mathbf{v}_k^{i_k}) - I_f(\mathbf{v}_k)}{v_i f(|\mathbf{v}_k|)},$$

where we again denote by $\mathbf{v}_k^{i_k}$ the vector that arises from $\mathbf{v}_k$ by moving the i_kth entry to the end of the vector. We may again apply the union Lemma (Lemma 6) to derive that this ratio is maximal when $i+1 = n$, i.e., there is only one entry after v_i. We then obtain

$$\frac{I_f(\mathbf{v}_k^{i_k}) - I_f(\mathbf{v}_k)}{v_i f(|\mathbf{v}_k|)} = \frac{yf(y+z) + xf(x+z) - xf(x+z) - yf(x+y+z)}{xf(x+y+z)},$$

which then establishes the result. ◀

Symmetrically to the previous corollary, we use the Split Lemma (Lemma 7) to derive closed form expressions for the two missing cases of $\hat{\mu}(f)$ for convex f and $\check{\mu}(f)$ for concave f. It is interesting to note that by repeatedly applying the Split Lemma the discrete integral approaches the integral, in the supremum.

▶ **Corollary 9.** *For a function $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, let*

$$A_f(x,y,z) = \frac{\int_z^y f(s)\,\mathrm{d}s + xf(x+y+z) - \int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s - xf(x+z)}{xf(x+y+z)}.$$

If f is convex, then $\hat{\mu}(f) = \sup_{x,y\in\mathbb{R}_{>0}, z\in\{0\}\cup[x,\infty)} A_f(x,y,z)$, and if f is concave then $\check{\mu}(f) = \sup_{x,y\in\mathbb{R}_{>0}, z\in\{0\}\cup[x,\infty)} -A_f(x,y,z)$.

Proof. We start to prove $A(f) = \hat{\mu}(f)$, if f is convex. First, recall that $\hat{\mu}(f)$ is defined as

$$\hat{\mu}(f) = \sup_{\mathbf{v}\in\mathbb{R}^n_{>0}} \max_{i\in\{1,\dots,n\}} \frac{I_f(\hat{\sigma}^i_f(\mathbf{v})) - I_f(\hat{\sigma}_f(\mathbf{v}))}{v_i f(|\mathbf{v}|)}.$$

By Lemma 5, it is without loss of generality to assume that $\hat{\sigma}_f(\mathbf{v})$ sorts the entries of $\mathbf{v}$ in non-increasing order. This implies that there are sequences $(n_k)_{k\in\mathbb{N}}$, $n_k \in \mathbb{N}$ for all $k \in \mathbb{N}$, $(i_k)_{k\in\mathbb{N}}$, $i_k \in \{1,\dots,n_k\}$ for all $k \in \mathbb{N}$ and a sequence of non-increasing vectors $(\mathbf{v}_k)_{k\in\mathbb{N}}$, $\mathbf{v}_k \in \mathbb{R}^{n_k}_{>0}$ such that

$$\lim_{k\to\infty} \frac{I_f(\mathbf{v}_k^{i_k}) - I_f(\mathbf{v}_k)}{v_{i_k} f(|\mathbf{v}_k|)} = \hat{\mu}(f).$$

where we denote by $\mathbf{v}_k^{i_k}$ the vector that arises from $\mathbf{v}_k$ by moving the i_kth entry to the end of the vector. For ease of exposition, let us set $n = n_k$, $i = i_k$, and $\mathbf{v} = (v_1,\dots,v_n) = \mathbf{v}_k$, for some fixed $k \in \mathbb{N}$. Let $z = \sum_{j=1}^{i-1} v_j$ denote the sum of the entries in $\mathbf{v}$ before entry v_i, let $x = v_i$, and let $y = \sum_{j=i+1}^{n}$ denote the sum of the entries in $\mathbf{v}$ after entry v_i. As $\mathbf{v}$ is non-increasing, we have $z \in \{0\} \cup [x,\infty)$. As shown Lemma 7, splitting all entries v_j, $j > i$ into two entries $v_j/2$ does only increase the difference of the discrete integrals. Applying the procedure repeatedly on all entries v_j, $j > i$, their contribution to the discrete integral approaches the integral. We obtain

$$\frac{I_f(\mathbf{v}^i) - I_f(\mathbf{v})}{v_i f(|\mathbf{v}|)} \le \frac{\int_z^y f(s)\,\mathrm{d}s + xf(x+y+z) - \int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s - xf(x+z)}{xf(x+y+z)}.$$

Applying this reasoning for all vectors $\mathbf{v}_k$, $k \in \mathbb{N}$, we obtain

$$\begin{aligned}&\sup_{\mathbf{v}\in\mathbb{R}^n_{>0}} \max_{i\in\{1,\dots,n\}} \frac{I_f(\mathbf{v}^i) - I_f(\mathbf{v})}{v_i f(|\mathbf{v}|)} \\ &\quad= \sup_{\substack{x,y\in\mathbb{R}_{>0}\\ z\in\{0\}\cup[x,\infty)}} \frac{\int_z^y f(s)\,\mathrm{d}s + xf(x+y+z) - \int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s - xf(x+z)}{xf(x+y+z)},\end{aligned}$$

as claimed.

For a concave function f, we obtain along the same lines

$$\lim_{k\to\infty} \frac{I_f(\mathbf{v}_k) - I_f(\mathbf{v}_k^{i_k})}{v_{i_k} f(|\mathbf{v}_k|)} = \breve{\mu}(f)$$

for some sequences $(n_k)_{k\in\mathbb{N}}$, $n_k \in \mathbb{N}$ for all $k \in \mathbb{N}$, $(i_k)_{k\in\mathbb{N}}$, $i_k \in \{1,\dots,n_k\}$ for all $k \in \mathbb{N}$ and a sequence of non-increasing vectors $(\mathbf{v}_k)_{k\in\mathbb{N}}$, $\mathbf{v}_k \in \mathbb{R}^{n_k}_{>0}$. We here again denote by $\mathbf{v}_k^{i_k}$ the vector that arises from $\mathbf{v}_k$ by moving the i_kth entry to the end of the vector. Repeated application of the Split Lemma (Lemma 7) gives

$$\breve{\mu}(f) = \frac{\int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s + xf(x+z) - \int_z^y f(s)\,\mathrm{d}s - xf(x+y+z)}{xf(x+y+z)},$$

analogous to the convex case. ◀

We proceed by further simplifying the computation of $\hat{\mu}(f)$ and $\breve{\mu}(f)$.

▶ **Lemma 10.** *Let $f : \mathbb{R}_{\geq 0} \to \mathbb{R}$ be a function. Then,*

$$\sup_{\substack{x,y\in\mathbb{R}_{>0},\\ z\in\{0\}\cup[x,\infty)}} A_f(x,y,z) = \sup_{x,y\in\mathbb{R}_{>0}} A_f(x,y,0), \qquad \text{if } f \text{ is convex,}$$

$$\sup_{\substack{x,y\in\mathbb{R}_{>0},\\ z\in\{0\}\cup[x,\infty)}} -A_f(x,y,z) = \sup_{x,y\in\mathbb{R}_{>0}} -A_f(x,y,0), \qquad \text{if } f \text{ is concave.}$$

Proof. We first prove the result for convex f. Let $(x_k)_{k\in\mathbb{N}}$, $(y_k)_{k\in\mathbb{N}}$, and $(z_k)_{k\in\mathbb{N}}$ sequences for which the supremum of $A_f(x,y,z)$ is attained. For fixed $k \in N$, let us set $x = x_k$, $y = y_k$, and $z = z_k$ and let $x' = x + z$, $y' = y$. We claim that

$$\begin{aligned} A_f(x,y,z) &= \frac{\int_z^{y+z} f(s)\,\mathrm{d}s + xf(x+y+z) - \int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s - xf(x+z)}{xf(x+y+z)} \\ &\leq \frac{\int_0^{y'} f(s)\,\mathrm{d}s + x'f(x'+y') - \int_{x'}^{x'+y'} f(s)\,\mathrm{d}s - x'f(x')}{x'f(x'+y')} = A_f(x'_k, y'_k, 0). \end{aligned} \tag{5}$$

Using $f(x'+y') = f(x+y+z)$ and substituting x' and y', (5) is equivalent to

$$\begin{aligned} &\frac{1}{x}\int_z^{y+z} f(s)\,\mathrm{d}s + f(x+y+z) - \frac{1}{x}\int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s - f(x+z) \\ &\quad\leq \frac{1}{x+z}\int_0^{y} f(s)\,\mathrm{d}s + f(x+y+z) - \frac{1}{x+z}\int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s - f(x+z). \end{aligned} \tag{6}$$

Rearranging terms (6) is equivalent to

$$\frac{1}{x}\int_z^{y+z} f(s)\,\mathrm{d}s - \frac{1}{x}\int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s \leq \frac{1}{x+z}\int_0^{y} f(s)\,\mathrm{d}s - \frac{1}{x+z}\int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s. \tag{7}$$

Put differently, we need to show that the function $h : \mathbb{R}_{\geq 0} \to \mathbb{R}$ defined as

$$h(y) = \frac{1}{x}\int_z^{y+z} f(s)\,\mathrm{d}s - \Big(\frac{1}{x} - \frac{1}{x+z}\Big)\int_{x+z}^{x+y+z} f(s)\,\mathrm{d}s - \frac{1}{x+z}\int_0^{y} f(s)\,\mathrm{d}s$$

is non-positive on $\mathbb{R}_{\geq 0}$. It is easy to verify that this is true for $y = 0$. Thus it suffices to show that $h'(y) \leq 0$ for all $y \geq 0$. To this end, we derive

$$\begin{aligned} h'(y) &= \frac{1}{x}f(y+z) - \Big(\frac{1}{x} - \frac{1}{x+z}\Big)f(x+y+z) - \frac{1}{x+z}f(y) \\ &= \frac{1}{x}\Big(f(y+z) - \big(1 - \frac{x}{x+z}\big)f(x+y+z) - \frac{x}{x+z}f(y)\Big), \end{aligned}$$

which is non-positive due to the convexity of f. We conclude that for each x_k, y_k, and z_k in the sequence that goes to the supremum of $A_f(x_k, y_k, z)$, there are x'_k and y'_k with $A_f(x_k, y_k, z_k) \leq A_f(x'_k, y'_k, 0)$. This implies the result for convex f.

To see the statement for concave f, note that since we seek to compute the supremum of $-A_f(x,y,z)$ rather than $A_f(x,y,z)$ all inequality signs in (5), (6), and (7) reverse and we shall show that the function h is non-negative on $\mathbb{R}_{\geq 0}$. It is easy to check that $h(y) = 0$ and $h'(y) \geq 0$ due to the concavity of f. This implies the result. ◀

▶ **Lemma 11.** *Let $f : \mathbb{R}_{\geq 0} \to \mathbb{R}$ be a function. Then,*

$$\sup_{\substack{x,y,z\in\mathbb{R}_{>0},\\ y\geq x}} B_f(x,y,z) = \sup_{x,y\in\mathbb{R}_{>0}} B_f(x,y,0), \qquad \text{if } f \text{ is convex,}$$

$$\sup_{\substack{x,y,z\in\mathbb{R}_{>0},\\ y\geq x}} -B_f(x,y,z) = \sup_{x,y\in\mathbb{R}_{>0}} -B_f(x,y,0), \qquad \text{if } f \text{ is concave.}$$

Proof. We first prove the result for convex f. Let $(x_k)_{k\in\mathbb{N}}$, $(y_k)_{k\in\mathbb{N}}$, and $(z_k)_{k\in\mathbb{N}}$ be sequences for which the supremum of B_f is attained in the limit. For fixed $k \in \mathbb{N}$, let us set $x = x_k$, $y = y_k$, and $z = z_k$ and let $x' = x$ and $y' = y + z$. We claim that

$$\begin{aligned} B_f(x,y,z) &= \frac{xf(x+z) + yf(x+y+z) - yf(y+z) - xf(x+y+z)}{xf(x+y+z)} \\ &\leq \frac{x'f(x') + y'f(x'+y') - y'f(y') - x'f(x'+y')}{x'f(x'+y')} = B_f(x',y',0) \end{aligned} \tag{8}$$

Substituting x' and y', (8) is equivalent to

$$xf(x+z) \leq xf(x) + zf(x+y+z) - zf(y+z) \tag{9a}$$

$$\Leftrightarrow \qquad \frac{f(x+z)-f(x)}{z} \leq \frac{f(x+y+z)-f(y+z)}{x}, \tag{9b}$$

which is satisfied since f is convex and $y \geq x$. For concave f, all inequality signs reverse. ◀

Combining all lemmas proven in this section, we obtain the following expressions for the computation of $\hat{\mu}(f)$ and $\check{\mu}(f)$, respectively.

▶ **Theorem 12.** *For a function $f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$. Then,*

$$\hat{\mu}(f) = \begin{cases} \sup\limits_{x,y\in\mathbb{R}_{>0}} \dfrac{\int_0^y f(s)\,\mathrm{d}s + xf(x+y) - \int_x^{x+y} f(s)\,\mathrm{d}s - xf(x)}{xf(x+y)}, & \text{if } f \text{ is convex} \\ \sup\limits_{\substack{x,y\in\mathbb{R}_{>0} \\ y\geq x}} \dfrac{yf(y) + xf(x+y) - xf(x) - yf(x+y)}{xf(x+y)}, & \text{if } f \text{ is concave,} \end{cases}$$

$$\check{\mu}(f) = \begin{cases} \sup\limits_{\substack{x,y\in\mathbb{R}_{>0} \\ y\geq x}} \dfrac{xf(x) + yf(x+y) - yf(y) - xf(x+y)}{xf(x+y)}, & \text{if } f \text{ is convex} \\ \sup\limits_{x,y\in\mathbb{R}_{>0}} \dfrac{\int_x^{x+y} f(s)\,\mathrm{d}s + xf(x) - \int_0^y f(s)\,\mathrm{d}s - xf(x+y)}{xf(x+y)}, & \text{if } f \text{ is concave.} \end{cases}$$

5 Polynomial Cost Functions

In this section, we consider the special case of polynomial functions. Formally, for $\ell \in \mathbb{N}$, let us denote by

$$\mathcal{C}^\ell = \left\{ f : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0} \mid \exists n \in \mathbb{N} \text{ with } f(x) = \sum_{j=1}^{n} a_j\, x^{b_j}, a_j \geq 0, b_j \in [1,\ell]\ \forall j \in \{1,\dots,n\} \right\}$$

the set of convex polynomial functions with non-negative coefficients and maximal degree ℓ. Combining Theorem 3 and Theorem 12, we show that weighted congestion with degree ℓ polynomials have an $(\ell+1)$-approximate pure Nash equilibrium.

▶ **Theorem 13.** *Every weighted congestion game with cost functions in $\mathcal{C}^\ell$ has an α-approximate pure Nash equilibrium with $\alpha \leq \ell + 1$.*

Proof. We first decompose the game into an isomorphic weighted congestion game G in which each resource r has a cost function of type $c_r(x) = a_r x^{b_r}$ with $a_r > 0$ and $b_r \in [1,\ell]$. Let us denote the set of these functions by $\bar{\mathcal{C}}^\ell$.

Theorem 3 shows that G has an $(1+\check{\mu})$-approximate pure Nash equilibrium and Theorem 12 provides an explicit formula for computing $\check{\mu}$. We can thus compute the maximum approximation factor α as

$$\begin{aligned}\alpha &= \sup_{f\in\bar{\mathcal{C}}^\ell}\sup_{\substack{x,y\in\mathbb{R}_{>0}\\ y\geq x}} \frac{xf(x)+yf(x+y)-yf(y)-xf(x+y)}{xf(x+y)}+1\\ &= \sup_{b\in[1,\ell]}\sup_{\substack{x,y\in\mathbb{R}_{>0}\\ y\geq x}} \frac{x^{b+1}+y(x+y)^b-y^{b+1}-x(x+y)^b}{x(x+y)^b}+1\\ &= \sup_{b\in[1,\ell]}\sup_{\substack{x,y\in\mathbb{R}_{>0}\\ y\geq x}} \frac{y\big((x+y)^b-y^b\big)-x\big((x+y)^b-x^b\big)}{x(x+y)^b}+1\end{aligned}$$

We use that for all $b\in[1,\ell]$ the function $s\mapsto s^b$ is convex with derivative $s\mapsto bs^{b-1}$ and obtain

$$\alpha \leq \sup_{b\in[1,\ell]}\sup_{\substack{x,y\in\mathbb{R}_{>0}\\ y\geq x}} \frac{xy\big(b\cdot(x+y)^{b-1}\big)-xy\big(bx^{b-1}\big)}{x(x+y)^b}+1.$$

We simplify and obtain

$$\alpha \leq \sup_{b\in[1,\ell]}\sup_{\substack{x,y\in\mathbb{R}_{>0}\\ y\geq x}} b\cdot\Big(\frac{y}{x+y}-\frac{yx^{b-1}}{(x+y)^b}\Big)+1 \leq \sup_{b\in[1,\ell]} b+1=\ell+1,$$

which completes the proof. ◀

For low degrees, we use the minimizing permutation instead of the maximizing permutation and obtain better bounds.

▶ **Theorem 14.** *Every weighted congestion game with cost functions in $\mathcal{C}^\ell$ has an α-approximate pure Nash equilibrium, where*

$$\alpha = \sup_{b\in[1,\ell]}\sup_{x,y\in\mathbb{R}_{>0}} \frac{\lambda(1+\lambda)^b}{\frac{1}{b+1}(1+\lambda)^{b+1}+(1-\frac{1}{b+1})\lambda^{b+1}-\frac{1}{b+1}}. \tag{10}$$

Proof. Using the same decomposition argument as in the proof of Theorem 13, we can assume without loss of generality that all cost functions are monomials. We again denote the set of monomials with maximum degree ℓ by $\bar{\mathcal{C}}^\ell$. Using Theorem 4 and Theorem 12, we obtain

$$\begin{aligned}\alpha &\leq \sup_{f\in\bar{\mathcal{C}}^\ell}\sup_{x,y\in\mathbb{R}_{>0}} \frac{xf(x+y)}{\int_x^{x+y}f(s)\,\mathrm{d}s+xf(x)-\int_0^y f(s)\,\mathrm{d}s}\\ &\leq \sup_{b\in[1,\ell]}\sup_{x,y\in\mathbb{R}_{>0}} \frac{x(x+y)^b}{\frac{1}{b+1}(x+y)^{b+1}+(1-\frac{1}{b+1})x^{b+1}-\frac{1}{b+1}y^{b+1}}\end{aligned}$$

Substituting $x=\lambda y$ for some $\lambda\in\mathbb{R}_{>0}$, we obtain

$$\alpha \leq \sup_{b\in[1,\ell]}\sup_{\lambda\in\mathbb{R}_{>0}} \frac{\lambda(1+\lambda)^b}{\frac{1}{b+1}(1+\lambda)^{b+1}+(1-\frac{1}{b+1})\lambda^{b+1}-\frac{1}{b+1}},$$

as claimed. ◀

For polynomials with small maximal degrees ℓ, we solve (10) and obtain approximation guarantees strictly below the factor of $\ell+1$ guaranteed by Theorem 13. Specifically, we obtain a factor of 4/3 for (at most) quadratic cost functions, 1.785 for (at most) cubic cost functions, and a factor of 2.326 for polynomials with maximum degree 4.

6 Concave Cost Functions

As the main result of this section, we shown that weighted congestion games with concave cost functions have a 3/2-approximate pure Nash equilibrium.

▶ **Theorem 15.** *Every weighted congestion game with concave cost functions has* 3/2*-approximate pure Nash equilibrium.*

To prove this theorem we show that $\breve{\mu}(f) \leq 1/2$ for each concave function f. Due to space constraints the detailed proof is deferred to the full version of this paper.

7 Two-player Games

In the following we analyze two-player weighted congestion games and provide a tight upper bound for approximate equilibria. For this section, it is convenient to assume that the set of cost functions $\mathcal{C}$ is closed under addition, i.e, $f \in \mathcal{C}$ and $g \in \mathcal{C}$ implies $(f+g) \in \mathcal{C}$. However, it is straightforward to extend our results to sets of cost functions that do not have this property.

As a lower bound, consider the following game with two players $N = \{1, 2\}$ with demands $x, y > 0$, respectively, and resources $R = \big\{(0,0), (0,1), (1,0), (1,1)\big\}$ with symmetric cost functions $c_{(0,0)} = c_{(1,1)} = c_1(x)$ and $c_{(1,0)} = c_{(0,1)} = c_2(x)$ for some $c_1, c_2 \in \mathcal{C}$. Player 1 has the choice between the strategies $\{(0,0),(0,1)\}$ and $\{(1,0),(1,1)\}$ whereas the strategies for player 2 are given by $\{(0,0),(1,0)\}$ and $\{(0,1),(1,1)\}$. For this game not to have an α-approximate NE it is necessary for the players to be able to take turns in improving their private cost by factors of at least α. We can therefore bound the value of α from above:

$$\alpha \leq \min\left\{\frac{c_2(x+y)+c_1(x)}{c_1(x+y)+c_2(x)}, \frac{c_1(x+y)+c_2(y)}{c_2(x+y)+c_1(y)}.\right\}$$

Clearly, for a given set of cost functions $\mathcal{C}$ and for every $\epsilon > 0$, by optimizing over $c_1, c_2 \in \mathcal{C}$ and $x, y \in \mathbb{R}_{>0}$, we can construct a game with no $(\alpha - \epsilon)$-approximate pure Nash equilibrium, where

$$\alpha = \sup_{x,y>0, c_1,c_2 \in \mathcal{C}} \min\left\{\frac{c_2(x+y)+c_1(x)}{c_1(x+y)+c_2(x)}, \frac{c_1(x+y)+c_2(y)}{c_2(x+y)+c_1(y)}\right\}.$$

As the main result of this section, we show that this bound tight. Due to space constraints the detailed proof is deferred to the full version of this paper.

▶ **Theorem 16.** *Every two-player weighted congestion game with cost functions in $\mathcal{C}$ has an α-approximate pure Nash equilibrium with*

$$\alpha \leq \sup_{x,y>0, c_1,c_2 \in \mathcal{C}} \min\left\{\frac{c_2(x+y)+c_1(x)}{c_1(x+y)+c_2(x)}, \frac{c_1(x+y)+c_2(y)}{c_2(x+y)+c_1(y)}\right\}.$$

For specific classes of cost functions, we solve for α and obtain the concrete numerical approximation factors shown in Table 1.

References

1 Heiner Ackermann, Heiko Röglin, and Berthold Vöcking. On the impact of combinatorial structure on congestion games. *Journal of the ACM*, 55(6), 2008.

2 Heiner Ackermann, Heiko Röglin, and Berthold Vöcking. Pure Nash equilibria in player-specific and weighted congestion games. *Theoretical Computer Science*, 410(17):1552–1563, 2009.

3 Heiner Ackermann and Alexander Skopalik. Complexity of pure Nash equilibria in player-specific network congestion games. *Internet Mathematics*, 5(4):323–342, 2008.

4 Ioannis Caragiannis, Angelo Fanelli, Nick Gravin, and Alexander Skopalik. Efficient computation of approximate pure Nash equilibria in congestion games. In *Proceedings of the 52nd Annual Symposium on Foundations of Computer Science, (FOCS)*, pages 532–541, 2011.

5 Ioannis Caragiannis, Angelo Fanelli, Nick Gravin, and Alexander Skopalik. Approximate pure Nash equilibria in weighted congestion games: existence, efficient computation, and structure. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC)*, pages 284–301, 2012.

6 Ho-Lin Chen and Tim Roughgarden. Network design with weighted players. In *Proceedings of the 18th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 29–38, 2006.

7 Steve Chien and Alistair Sinclair. Convergence to approximate Nash equilibria in congestion games. In *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 169–178, 2007.

8 George Christodoulou, Elias Koutsoupias, and Paul G. Spirakis. On the performance of approximate equilibria in congestion games. In *Proceedings of the 17th Annual European Symposium on Algorithms (ESA)*, pages 251–262. Springer, 2009.

9 Juliane Dunkel and Andreas S. Schulz. On the complexity of pure-strategy Nash equilibria in congestion and local-effect games. In *Proceedings of the 2nd International Workshop Internet & Network Economics (WINE)*, pages 62–73, 2006.

10 Alex Fabrikant, Christos Papadimitriou, and Kunal Talwar. The complexity of pure Nash equilibria. In *Proceedings of the 36th ACM Symposium on Theory of Computing (STOC)*, pages 604–612, 2004.

11 Dimitris Fotakis, Spyros Kontogiannis, and Paul G. Spirakis. Selfish unsplittable flows. *Theoretical Computer Science*, 348(2-3):226–239, 2005.

12 Irving L. Glicksberg. A further generalization of the Kakutani fixed point theorem, with application to Nash equilibrium points. *Proceedings of the AMS*, 3:170–174, 1952.

13 Michel X. Goemans, Vahab S. Mirrokni, and Adrian Vetta. Sink equilibria and convergence. In *Proceedings of the 46th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 142–154, 2005.

14 Tobias Harks and Max Klimm. On the existence of pure Nash equilibria in weighted congestion games. *Mathematics of Operations Research*, 37(3):419–436, 2012.

15 Lavy Libman and Ariel Orda. Atomic resource sharing in noncooperative networks. *Telecommunication Systems*, 17(4):385–409, 2001.

16 Igal Milchtaich. Congestion games with player-specific payoff functions. *Games and Economic Behavior*, 13(1):111–124, 1996.

17 John F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences, USA*, 36:48–49, 1950.

18 Panagiota N. Panagopoulou and Paul G. Spirakis. Algorithms for pure Nash equilibria in weighted congestion games. *Journal of Experimental Algorithmics*, 11, 2007.

19 Robert Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2:65–67, 1973.

20 Alexander Skopalik and Berthold Vöcking. Inapproximability of pure Nash equilibria. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 355–364, 2008.

Discrepancy Without Partial Colorings

Nicholas J. A. Harvey*[1], Roy Schwartz[2], and Mohit Singh[2]

1 **Department of Computer Science, University of British Columbia**
nickhar@cs.ubc.ca
2 **Microsoft Research, Redmond, WA**
{schwartz.roi,mohits}@gmail.com

Abstract

Spencer's theorem asserts that, for any family of n subsets of ground set of size n, the elements of the ground set can be "colored" by the values ± 1 such that the sum of every set is $O(\sqrt{n})$ in absolute value. All existing proofs of this result recursively construct "partial colorings", which assign ± 1 values to half of the ground set. We devise the first algorithm for Spencer's theorem that directly computes a coloring, without recursively computing partial colorings.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.1.6 Optimization, G.2.1 Combinatorics

Keywords and phrases Combinatorial Discrepancy, Brownian Motion, Semi-Definite Programming, Randomized Algorithm

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.258

I cannot pretend to feel impartial about colours. I rejoice with the brilliant ones and am genuinely sorry for the poor browns.

— Winston Churchill

1 Introduction

In combinatorics, the *discrepancy* problem can be stated as follows. Given a universe $U = \{1, 2, \ldots, n\}$ and a family of subsets $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ the goal is to find a function $\chi : U \to \{\pm 1\}$ that minimizes

$$\max_{1 \le j \le m} \left\{ \left| \sum_{i \in S_j} \chi(i) \right| \right\} . \tag{1}$$

The function χ is called a *coloring*. The *discrepancy* of $\mathcal{S}$ is the minimum of (1) over all colorings. Determining the discrepancy of a set system is a fundamental problem in combinatorics [1, 7, 14] that has a wide range of applications in computer science [9, 5, 10, 15]. One of the most celebrated results in this area is Spencer's theorem [17] stating that any family $\mathcal{S}$ with $m = n$ has discrepancy at most $6\sqrt{n}$. More generally, if $m \ge n$, the upper bound becomes $O\left(\sqrt{n \cdot \log\left((2m)/n\right)}\right)$. This bound is tight up to constant factors for all $m \ge n$. Recently, efficient algorithms were developed [4, 3, 13] to construct colorings that match Spencer's bounds up to constant factors.

* Supported by an NSERC Discovery Grant and a Sloan Foundation Fellowship.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 258–273

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

Discrepancy is also a topic of major interest in convex geometry, and many combinatorial discrepancy results have a more general geometric statement. The geometric form of Spencer's theorem is: for all $\{x_1, \ldots, x_n\} \subset [-1,1]^n$, there exists $\chi : U \to \{\pm 1\}$ with $\left\|\sum_{i \in U} x_i \chi(i)\right\|_\infty \leq 6\sqrt{n}$. This geometric form follows from Spencer's original proof, and it is also a special case of a geometric result that was independently proven by Gluskin [12]. Gluskin's proof was simplified by Giannopoulos [11], and an algorithmic form of Giannopoulos' theorem was recently given by Rothvoss [16].

All of the previous work on Spencer's theorem, including the geometric results and the algorithmic results, are based on the idea of producing a *partial coloring*. In this approach, one first obtains a coloring of half the elements of U, then recurses on the residual family of subsets obtained by deleting all colored elements. Although the partial coloring approach suffices to obtain tight results for Spencer's theorem, there are other important discrepancy problems for which this approach does not currently (and perhaps cannot) yield tight results. A notable example is the Beck-Fiala conjecture [6], which asserts that: for every set system $\mathcal{S}$ for which every element of U is contained in at most t sets, the discrepancy of $\mathcal{S}$ is $O(\sqrt{t})$. The geometric form of the Beck-Fiala conjecture is the Komlós conjecture: for all $\{x_1, \ldots, x_n\} \subset \mathbb{R}^n$ with $\|x_i\|_2 \leq 1$, there exists $\chi : U \to \{\pm 1\}$ with $\left\|\sum_{i \in U} x_i \chi(i)\right\|_\infty \leq O(1)$.

All known results [18, 4, 13] towards these conjectures that are based on partial coloring have the drawback that they incur an extra factor of $O(\log n)$ in the discrepancy, due to the $O(\log n)$ recursive steps. The only known approach for these conjectures that avoids the extra $O(\log n)$ factor is Banaszczyk's geometric technique [2], which is not based on partial coloring, and incurs only an $O(\sqrt{\log n})$ factor, but has the drawback that it is not algorithmic. Due to the drawbacks of these previous results, it has been an open question to find new techniques that avoid partial colorings for these discrepancy problems, particularly algorithmic techniques. Such new techniques would hopefully lead to progress on the Beck-Fiala/Komlós conjectures.

1.1 Our Contribution

In this work we devise the first algorithm for Spencer's theorem that directly computes a (full) coloring, without recursively computing partial colorings. Our algorithm builds upon the techniques of Bansal [4] and Lovett and Meka [13], which we now review.

Let $c_1, \ldots, c_m$ be suitable parameters, and define

$$\mathcal{P}^{\text{disc}} \;=\; \left\{\mathbf{x} \in \mathbb{R}^n : \left|\mathbf{1}_{S_j} \cdot \mathbf{x}\right| \leq c_j \;\; \forall 1 \leq j \leq m\right\}.$$

Bansal's breakthrough result [4] performs a random walk with Gaussian increments (i. e., discretized Brownian motion) starting at the origin. The covariance matrix of each Gaussian step comes from a feasible solution to a semidefinite program (SDP) that describes a "vector relaxation" of the discrepancy problem. If at any time the random walk approaches a face of $[-1,1]^n$, it sticks to that face and continues walking within that face. If at any time the random walk gets very close to a discrepancy constraint, i. e., a face of $\mathcal{P}^{\text{disc}}$, that discrepancy constraint is pushed away from the origin to very carefully chosen distances, and the SDP is modified accordingly. Spencer's non-constructive theorem is used to ensure feasibility of each SDP.

Lovett and Meka [13] perform a similar random walk, except that every Gaussian step has covariance matrix equal to the identity. If at any time the random walk gets very close to a discrepancy constraint, it sticks to that face and continues walking within that face. They prove that, when the random walk stops, a constant fraction of the elements are colored, thus obtaining a partial coloring.

Both of these algorithms necessarily result in a partial coloring, not a full coloring. For Bansal's algorithm, this is because feasibility of the SDP is proven using Spencer's theorem, which only ensures existence of a "partial vector coloring", not a "full vector coloring". For the Lovett-Meka algorithm, this is because their random walk will likely stick to many discrepancy constraints before terminating. The intersection of these discrepancy constraints need not contain any point in $\{-1,1\}^n$, and hence the walk cannot directly produce a full coloring.

Our algorithm borrows many ideas from Bansal and from Lovett-Meka, but has two key differences.

- The first difference is the way in which we distort the random walk. Like Bansal, our Gaussian steps may use different covariance matrices. Whereas Bansal's covariance matrices change only when the SDP changes (i. e., when the walk gets very close to a discrepancy constraint), our walk's covariance matrices change in every step. Thus, our walk should be viewed as a discretized diffusion process. Our covariance matrices do not directly come from vector colorings, but instead from a more geometric viewpoint. We prove the following geometric result: for any polytope $\mathcal{P}$ and point $\theta \in \mathcal{P}$ there exists an ellipsoid centered at θ and contained inside $\mathcal{P}$ such that the trace of the semi-definite matrix defining the ellipsoid is large compared to the distances of the closest faces of $\mathcal{P}$ to θ. Our proof of this geometric claim uses SDP duality and might be of independent interest. Unfortunately this geometric approach by itself is not sufficient, as one might end up close to a vertex of $\mathcal{P}^{\text{disc}} \cap [-1,1]^n$, thus getting stuck without the ability to fully color all the elements.
- The second difference is that we slowly move all discrepancy constraints away from the origin in every step. This allows the random walk to escape potential areas of $\mathcal{P}^{\text{disc}}$ in which it might get stuck. Furthermore, the distance that every constraint is moved is a deterministic function of the the time step, whereas in Bansal's approach the movement of the constraints depends on the random walk. The movement of our constraints must also be carefully chosen. On one hand the rate in which the discrepancy constraints are pushed needs to be slow enough such that one still obtains optimal discrepancy bounds. On the other hand this rate needs to be fast enough so the random walk does not get stuck too close to some discrepancy constraints. At the heart of this approach is the following simple observation: the variance of the distance between the location of the random walk and any fixed discrepancy constraint is upper bounded by the number of elements which are still uncolored. Hence, a delicate balance is needed to ensure that the rate in which the discrepancy constraints are moved is larger than the number of uncolored elements.

Let us make one final remark concerning the difference between our algorithm and previous ones. As stated above, our algorithm directly computes a full coloring without recursively computing partial colorings. On the other hand, our *analysis* partitions the algorithm's execution into several phases, which are somewhat analogous to the partial coloring steps of the Bansal and Lovett-Meka algorithms. Nevertheless, it is still accurate to say that our algorithm does not use partial colorings as the algorithm's behavior is oblivious to these phases of the analysis.

2 Preliminaries

Ellipsoids: Given a positive semi-definite $n \times n$ matrix Σ and its symmetric $n \times n$ square root matrix B (i. e. $\Sigma = B^2$) we denote the ellipsoid it defines centered at point $\theta \in \mathbb{R}^n$ by:

$$E(\Sigma, \theta) \triangleq \{\mathbf{y} = B \cdot \mathbf{u} + \theta : \mathbf{u} \in \mathbb{R}^n, ||\mathbf{u}||_2 \leq 1\} \ .$$

Additionally, we denote the Euclidean sphere centered at θ with radius r by: $\text{Ball}(\theta, r)$.

The Discrepancy Polytope: We denote by $\mathbf{1}_S \in \mathbb{R}^n$ the characteristic vector of the subset of elements $S \subseteq U$. Our algorithm conducts iterations which we index by t. For any subset $S_j \in \mathcal{S}$ and iteration t we define:

$$c_j(t) \triangleq C \cdot \sqrt{n \cdot \ln\left(\frac{2m}{n}\right)} \cdot \left(1 - 2^{-\frac{\gamma^2 \cdot t}{a}}\right) \ .$$

Here C and a are absolute constants and γ a parameter depending on n, all to be chosen later. We define the following polytopes:

$$\mathcal{P}^{\text{disc}}(t) \triangleq \left\{\mathbf{x} \in \mathbb{R}^n : \forall 1 \leq j \leq m \quad \left|\mathbf{1}_{S_j} \cdot \mathbf{x}\right| \leq c_j(t)\right\}$$
$$\mathcal{P}(t) \triangleq \mathcal{P}^{disc}(t) \cap \{\mathbf{x} \in \mathbb{R}^n : \forall 1 \leq i \leq n \quad |x_i| \leq 1\} \ .$$

Distances: In order to define our notion of *effective* distance of a point $\theta \in \mathcal{P}(t)$ from the jth discrepancy constraint, we need the following definition of the set of variables which are active, i. e., all variables which are not colored:

$$\mathcal{C}^{\text{act}}(\theta) \triangleq \{1 \leq i \leq n : |\theta_i| < 1 - 1/n\} \ .$$

We also denote the set of active variables in $S \subseteq U$ by: $S^{\text{act}}(\theta) \triangleq S \cap \mathcal{C}^{\text{act}}(\theta)$. For every point $\theta \in \mathcal{P}$ we denote its *distance* with respect to the jth discrepancy constraint by:

$$d_j(\theta, t) \triangleq \frac{c_j(t) - \left|\mathbf{1}_{S_j} \cdot \theta\right|}{||\mathbf{1}_{S_j^{\text{act}}(\theta)}||_2} \ .$$

Gaussian Distribution and Concentration: If $X \sim N(0,1)$ we denote the cumulative distribution function of the normalized gaussian by: $\Phi(t) = \Pr[X \leq t]$. We also require the following concentration result.

▶ **Lemma 2.1** (Bansal [4]). *Let $X_1, \ldots, X_T$ be random variables and $Y_1, \ldots, Y_T$ be random variables where each Y_i is a function of X_i. Suppose that for all $1 \leq i \leq T$ and $x_1, \ldots, x_{i-1} \in \mathbb{R}$, $Y_i|_{X_1 = x_1, \ldots, X_{i-1} = x_{i-1}} \sim N(0, \rho(x_1, \ldots, x_{i-1}))$ where $\rho(x_1, \ldots, x_{i-1}) \leq 1$. Then for any $\lambda \geq 0$:*

$$\Pr\left[|Y_1 + \ldots Y_T| \geq \lambda\sqrt{T}\right] \leq 2 \cdot e^{-\lambda^2/2} \ .$$

3 Algorithm

In this section we present an algorithm that fully colors all elements without resorting to partial coloring. Our algorithm conducts a random walk where in each step the direction that the algorithm moves to is determined by a suitable ellipsoid. Specifically, given a point θ and time t, a maximum trace ellipsoid that is contained inside $\mathcal{P}^{\text{disc}}(t)$ is found. In

addition to being contained inside $\mathcal{P}^{\text{disc}}(t)$, we impose two additional requirements. The first is that the ellipsoid is also contained in the subspace of all variables that are still active: $\{\mathbf{x} \in \mathbb{R}^n : x_i = \theta_i \ \forall i \in U \setminus \mathcal{C}^{\text{act}}(\theta)\}$. The reason for that is that a variable i which is not active anymore is fully colored, i. e., $|x_i| \geq 1 - 1/n$, and its value should not be changed. The second requirement is that the ellipsoid is not too large and is in fact contained inside the Euclidean unit sphere centered at θ. Such an ellipsoid can be found, for example, by solving the following semi-definite program:

$$\begin{aligned} SDP(\theta, t) \qquad \max \quad & Tr\left(\Sigma\right) \\ s.t. \quad & E\left(\Sigma, \theta\right) \subseteq \mathcal{P}^{\text{disc}}(t) \cap \left\{\mathbf{x} \in \mathbb{R}^n : x_i = \theta_i \ \forall i \in U \setminus \mathcal{C}^{\text{act}}(\theta)\right\} \cap \text{Ball}(\theta, 1) \end{aligned}$$

Note that the above semi-definite program is parameterized by a point θ and time t. Let us now provide a precise description of our algorithm and our main theorem.

Algorithm 1: $(n, \mathcal{S}, \gamma)$

```
Initialize: x(0) ← 0, t ← 0, γ ← 1/n².
while C^act(x(t)) ≠ ∅ do
    Let B(x(t), t) be the square root of the solution for SDP(x(t), t).
    Choose g(t) ∈ ℝ^n s. t. g_i(t) ~ N(0, 1) i.i.d. ∀1 ≤ i ≤ n.
    x(t + 1) ← x(t) + γ · B(x(t), t) · g(t).
    if x(t + 1) ∉ P(t + 1) then
        Abort.
    t ← t + 1.
for i = 1 to n do
    Round x_i(t) to the closest integer.
Output x(t).
```

▶ **Theorem 3.1.** *With probability at least $1/poly(n)$ Algorithm 1 terminates in polynomial time without aborting and outputs a coloring with discrepancy of $O\left(\sqrt{n \cdot \ln\left(2m/n\right)}\right)$.*

Note: We remark that we can in fact compute a suitable ellipsoid without solving $SDP(\theta, t)$. It can be inferred from our proof techniques that one can directly compute a feasible solution to $SDP(\theta, t)$, whose objective value is sufficiently high that it is enough to guarantee the correctness of Theorem 3.1. This direct computation requires only the use of Gram-Schmidt orthogonalization, thus making Algorithm 1 considerably faster and simpler. Details are deferred to the full version of the paper.

4 Analysis

We first present the geometric core of our argument, namely that there is a suitable ellipsoid that Algorithm 1 can choose in every iteration t. Then we proceed by showing that this maximum trace ellipsoid is enough to prove the correctness of the algorithm as stated in Theorem 3.1.

4.1 Geometric Core

At the heart of our geometric approach lies the following theorem, which proves the existence of a suitable ellipsoid. Specifically, the ellipsoid we find is centered at a given point θ and is

contained inside the given polytope $\mathcal{P}$. The trace of the semi-definite matrix defining the ellipsoid is comparable to the distances of the closest faces of $\mathcal{P}$ to θ.

▶ **Theorem 4.1.** *Let* $\mathcal{P} = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{a}_i \cdot \mathbf{x} = b_i \ \forall 1 \leq i \leq k,\ \mathbf{v}_j \cdot \mathbf{x} \leq c_j \ \forall 1 \leq j \leq m\}$ *such that* $\mathbf{a}_i \cdot \mathbf{v}_j = 0$ *for each* $1 \leq i \leq k$ *and* $1 \leq j \leq m$. *Let* $\theta \in \mathcal{P}$ *and* $d_j(\theta) \triangleq \frac{c_j - \mathbf{v}_j \cdot \theta}{\|\mathbf{v}_j\|_2}$. *Then there exists an ellipsoid* $E(\Sigma, \theta) \subseteq \mathcal{P}$ *such that* $Tr(\Sigma) \geq \min_{J:|J|=n-k} \left\{\sum_{j \in J} d_j(\theta)^2\right\}$.

Proof. Using the fact that $E(\Sigma, \theta) \subseteq \mathcal{P}$ if and only the constraints in *(Primal-SDP)* are satisfied (see Chapter 8, page 428 [8]), we obtain that it is enough to show that the objective of the following semi-definite program is more than $\min_{J:|J|=n-k} \left\{\sum_{j \in J} d_j(\theta)^2\right\}$.

$$\begin{aligned}
\max \quad & Tr(\Sigma) && \textit{(Primal-SDP)} \\
s.t. \quad & \langle \mathbf{a}_i \mathbf{a}_i^T, \Sigma \rangle = 0 && \forall 1 \leq i \leq k \\
& \left(||\mathbf{v}_j||_2^2\right)^{-1} \langle \mathbf{v}_j \mathbf{v}_j^T, \Sigma \rangle \leq \left(||\mathbf{v}_j||_2^2\right)^{-1} (c_j - \mathbf{v}_j \cdot \theta)^2 = d_j(\theta)^2 && \forall 1 \leq j \leq m \\
& \Sigma \succeq 0
\end{aligned}$$

Consider the dual of *(Primal-SDP)*:

$$\begin{aligned}
\min \quad & \sum_{j=1}^{m} \lambda_j d_j(\theta)^2 && \textit{(Dual-SDP)} \\
s.t. \quad & \sum_{i=1}^{k} \mu_i \left(\mathbf{a}_i \mathbf{a}_i^T\right) + \sum_{j=1}^{m} \lambda_j \left(||\mathbf{v}_j||_2^2\right)^{-1} \left(\mathbf{v}_j \mathbf{v}_j^T\right) \succeq I \\
& \lambda_j \geq 0 && \forall 1 \leq j \leq m
\end{aligned}$$

By renaming the constraints assume, without loss of generality, that $d_1(\theta) \leq \ldots \leq d_m(\theta)$. We will show that the dual objective value for any feasible dual solution (λ, μ) is at least $\sum_{j=1}^{n-k} d_j(\theta)^2$ which will prove the theorem.

For any $0 \leq t \leq m$, consider the subspace S_t that is orthogonal to the vectors $\{\mathbf{a}_i\}_{i=1}^k$ and the vectors $\{\mathbf{v}_j\}_{j=1}^t$. Note that the dimension of S_t is at least $n - k - t$. Denote by B_t the matrix whose columns form an orthonormal basis of S_t. Taking the inner product of the dual constraint with $B_t B_t^T$, we obtain that:

$$\sum_{i=1}^{k} \mu_i \left(B_t B_t^T\right) \cdot \left(\mathbf{a}_i \mathbf{a}_i^T\right) + \sum_{j=1}^{m} \lambda_j \left(||\mathbf{v}_j||_2^2\right)^{-1} \left(B_t B_t^T\right) \cdot \left(\mathbf{v}_j \mathbf{v}_j^T\right) \geq \left(B_t B_t^T\right) \cdot I \ . \tag{2}$$

Let us focus first on the l.h.s. of (2). Note that for every $1 \leq i \leq k$:

$$\left(B_t B_t^T\right) \cdot \left(\mathbf{a}_i \mathbf{a}_i^T\right) = \|B_t^T \mathbf{a}_i\|_2^2 \overset{\text{(i)}}{=} 0 \ .$$

Equality (i) is derived from the fact that the columns of B_t are orthogonal to $\{\mathbf{a}_i\}_{i=1}^k$. Similarly, one can show that $\left(B_t B_t^T\right) \cdot \left(\mathbf{v}_j \mathbf{v}_j^T\right) = 0$ for any $1 \leq j \leq t$. Additionally, for any $t + 1 \leq j \leq m$, we have that:

$$\left(B_t B_t^T\right) \cdot \left(\mathbf{v}_j \mathbf{v}_j^t\right) = \|B_t^T \mathbf{v}_j\|_2^2 \overset{\text{(ii)}}{\leq} \|\mathbf{v}_j\|_2^2 \ .$$

Inequality (ii) follows since the columns of B_t form an orthonormal basis. Hence, we can conclude that the l.h.s. of (2) is upper bounded by $\sum_{j=t+1}^{m} \lambda_j$. Let us focus now on the r.h.s. of (2):

$$\left(B_t B_t^T\right) \cdot I = Tr(B_t B_t^T) = Tr(B_t^T B_t) \overset{\text{(iii)}}{\geq} n - k - t \ .$$

Inequality (iii) is derived from the fact that the columns of B_t are an orthonormal basis of dimension at least $n - k - t$. Thus, combining the upper bound on the l.h.s. of (2) and lower bound on the r.h.s. of (2), we obtain that for each $0 \leq t \leq m$:

$$\sum_{j=t+1}^{m} \lambda_j \geq n - k - t \ .$$

Our goal is to lower bound the value of any feasible solution for *(Dual-SDP)*. This can be done by considering the following linear program, whose variables are $\{\lambda_j\}_{j=1}^{m}$, and is a relaxation of *(Dual-SDP)*:

$$\begin{aligned} \min \quad & \sum_{j=1}^{m} \lambda_j d_j(\theta)^2 & \\ s.t. \quad & \sum_{j=t+1}^{m} \lambda_j \geq n - k - t & \forall 0 \leq t \leq m \\ & \lambda_j \geq 0 & \forall 1 \leq j \leq m \end{aligned}$$

Since *(Dual-SDP)* is a minimization problem, it suffices to lower bound the value of an optimal solution. Recall that $d_1(\theta) \leq \ldots \leq d_m(\theta)$ and all the λ_js are non-negative. Therefore, the optimal solution to the above linear program is $\lambda_j = 1$ for each $1 \leq j \leq n - k$ and $\lambda_j = 0$ for each $j > n - k$. Thus, we can conclude that $\sum_{j=1}^{m} \lambda_j d_j(\theta)^2 \geq \sum_{j=1}^{n-k} d_j(\theta)^2$ as claimed. ◀

Our analysis of Algorithm 1 actually requires the following corollary of Theorem 4.1. We choose the polytope $\mathcal{P}$ to correspond to the requirements we mentioned in Section 3: given a point θ, in addition to being contained in $\mathcal{P}^{\text{disc}}(t)$, the ellipsoid should also be contained in the subspace of all elements that are still active, i. e., $\{\mathbf{x} \in \mathbb{R}^n : x_i = \theta_i \ \forall i \in U \setminus \mathcal{C}^{\text{act}}(\theta)\}$, and the Euclidean unit sphere: $\text{Ball}(\theta, 1)$. The proof of the corollary appears in Appendix A.

▶ **Corollary 4.2.** *For every $t \geq 0$ and every $\theta \in \mathcal{P}(t)$, there exists an ellipsoid*

$$E(\Sigma, \theta) \subseteq \mathcal{P}^{disc}(t) \cap \left\{\mathbf{x} \in \mathbb{R}^n : x_i = \theta_i \ \forall i \in U \setminus \mathcal{C}^{act}(\theta)\right\} \cap Ball(\theta, 1)$$

that satisfies: $Tr(\Sigma) \geq \min_{J \subseteq \{1,\ldots,m\}:|J|=|\mathcal{C}^{act}(\theta)|} \left\{\sum_{j \in J} \min\left\{1, d_j(\theta, t)^2\right\}\right\}$. Moreover, Σ can be computed in polynomial time.

4.2 Phases

The analysis of Algorithm 1 is done in *phases*, each comprising of several consecutive iterations of the algorithm. Denote the sequence of t values indicating the starting iteration of the ith phase by τ_i, where $\tau_i = {}^{(b \cdot i)}/_{\gamma^2}$ for some absolute constant b to be chosen later. Specifically, the ith phase of Algorithm 1, where $i = 0, 1, 2, \ldots$, corresponds to the following t values:

$$\tau_i = \frac{b \cdot i}{\gamma^2} \leq t < \frac{b \cdot (i+1)}{\gamma^2} = \tau_{i+1} \ .$$

We require the notion of success, which is made formal in the following definition.

▶ **Definition 4.3.** Phase i is *successful* if at its end the algorithm has not aborted and:

$$\left|\mathcal{C}^{\text{act}}\left(\mathbf{x}\left(\tau_{i+1}\right)\right)\right| \leq 2^{-(i+1)}n \ .$$

We also require that the absolute constants C, a and b satisfy the following three conditions: $a \geq 8b$, $C \cdot \left(1 - 2^{-b/(2a)}\right) \geq \sqrt{32b}$, and $b \geq 64$. It is important to note that these conditions are *sufficient* for the correctness of Algorithm 1, but might not be necessary (they were chosen for simplicity of presentation alone). The main lemma we prove is the following.

▶ **Lemma 4.4.** *For every i, if phase $i-1$ is successful and $\gamma \leq b/n^2$, then phase i is successful with probability at least $1/4$.*

In order to prove Lemma 4.4 we start the analysis by showing that with overwhelming probability Algorithm 1 never aborts. The following lemma states that in each iteration there is an exponentially small probability of aborting. Its proof appears in Appendix B.

▶ **Lemma 4.5.** *For every iteration $t \geq 0$,*

$$\Pr\left[\mathbf{x}(t+1) \notin \mathcal{P}(t+1) | \mathbf{x}(t) \in \mathcal{P}(t)\right] \ \leq \ (2n+1) \cdot \left(1 - \Phi\left(\left(\gamma \cdot n\right)^{-1}\right)\right).$$

Moreover, if $\tau_i \leq t < \tau_{i+1}$, then the lemma holds also when conditioning that phase $i-1$ is successful.

Consider the following random subset:

$$A(t) \triangleq \left\{j : d_j\left(\mathbf{x}(t), t\right) \leq 1\right\} \ .$$

The random subset $A(t)$ consists of all discrepancy constraints j which are *bad*, as such constraints are close to the location of Algorithm 1 at time t, i. e., $\mathbf{x}(t)$. Let us lower bound the expected trace of the ellipsoid for every iteration t using $A(t)$.

▶ **Lemma 4.6.** *For every $t \geq 0$: $\mathbb{E}\left[Tr\left(B(\mathbf{x}(t), t)^2\right)\right] \geq \mathbb{E}\left[|\mathcal{C}^{act}(\mathbf{x}(t))|\right] - \mathbb{E}\left[|A(t)|\right]$. Moreover, if $\tau_i \leq t < \tau_{i+1}$, then the lemma holds also when conditioning that phase $i-1$ is successful.*

Proof.

$$\mathbb{E}\left[Tr\left(B\left(\mathbf{x}(t), t\right)^2\right)\right] \overset{\text{(i)}}{\geq} \mathbb{E}\left[\min_{J \subseteq \{1,\ldots,m\} : |J| = |\mathcal{C}^{\text{act}}(\mathbf{x}(t))|} \left\{\sum_{j \in J} \min\left\{1, d_j(\mathbf{x}(t), t)^2\right\}\right\}\right] \tag{3}$$

Inequality (i) is from Corollary 4.2. Let $J^*(t)$ be the random subset achieving the minimum value in the r.h.s. of (3). Then,

$$\begin{aligned}
&\mathbb{E}\left[\min_{J \subseteq \{1,\ldots,m\} : |J| = |\mathcal{C}^{\text{act}}(\mathbf{x}(t))|} \left\{\sum_{j \in J} \min\left\{1, d_j(\mathbf{x}(t), t)^2\right\}\right\}\right] \qquad (4)\\
&\quad = \mathbb{E}\left[\sum_{j \in J^*(t)} \min\left\{1, d_j(\mathbf{x}(t), t)^2\right\}\right] \geq \mathbb{E}\left[\sum_{j \in J^*(t) \setminus A(t)} \min\left\{1, d_j(\mathbf{x}(t), t)^2\right\}\right]\\
&\quad \overset{\text{(ii)}}{\geq} \mathbb{E}\left[|J^*(t) \setminus A(t)|\right] \geq \mathbb{E}\left[|J^*(t)|\right] - \mathbb{E}\left[|A(t)|\right]\\
&\quad \overset{\text{(iii)}}{=} \mathbb{E}\left[\left|\mathcal{C}^{\text{act}}(\mathbf{x}(t))\right|\right] - \mathbb{E}\left[|A(t)|\right]
\end{aligned}$$

Inequality (ii) follows from the fact that if $j \notin A(t)$ then $d_j(\mathbf{x}(t), t) \geq 1$ (by the definition of $A(t)$). Equality (iii) is true since $|J^*(t)| = |\mathcal{C}^{\text{act}}(\mathbf{x}(t))|$ by the definition of $J^*(t)$. Note that the exact same proof holds also when conditioning that phase $i-1$ is successful. ◀

It is clear that one wishes that $|A(t)|$ be as small as possible. The following lemma states that for many of the iterations of phase i, the expected size of $A(t)$ is indeed small enough. Its proof appears in Appendix C.

▶ **Lemma 4.7.** *For every $i \geq 0$ and iteration t, where $\tau_i + b/(2\cdot\gamma^2) \leq t < \tau_{i+1}$, if phase $i-1$ is successful then $\mathbb{E}\left[|A(t)|\right] \leq 2 \cdot 2^{-\frac{1}{8b}C^2\left(1-2^{-b/(2a)}\right)^2} \cdot \left(2^{-i}\cdot n\right)$.*

We are now ready to prove the main Lemma.

Proof of Lemma 4.4. For simplicity of presentation, we omit in this proof the notations indicating that all events and probabilities are conditioned on the event that phase $i-1$ is successful. First, let us bound the probability that Algorithm 1 does not abort during phase i. Using a union bound over all b/γ^2 iterations in phase i and applying Lemma 4.5, one can conclude that the probability of aborting during phase i is at most:

$$\frac{b}{\gamma^2}\cdot(2n+1)\cdot\left(1-\Phi\left(\frac{1}{\gamma\cdot n}\right)\right) .$$

Since $\gamma \leq b/n^2$, the application of standard gaussian tail bounds suffices to obtain a total aborting probability of at most $1/4$.

Second, let us prove that, at the end of phase i, $\Pr\left[|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_{i+1}))| > 2^{-(i+1)}\cdot n\right] < 1/2$. Note that this concludes the proof since one can apply a union bound over the latter event and the event that Algorithm 1 did not abort during phase i, yielding a failure probability of at most $3/4$.

We now examine two cases, depending on the value of $\mathbb{E}\left[|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_{i+1}))|\right]$. If we are in the case that $\mathbb{E}\left[|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_{i+1}))|\right] \leq 1/2\cdot\left(2^{-(i+1)}\cdot n\right)$ then Markov's inequality suffices. Otherwise, let us assume that $\mathbb{E}\left[|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_{i+1}))|\right] > 1/2\cdot\left(2^{-(i+1)}\cdot n\right)$. Note that $|\mathcal{C}^{\text{act}}(\mathbf{x}(t))|$ is a monotone non-increasing function in t, and therefore for every iteration t in phase i we have that: $\mathbb{E}\left[|\mathcal{C}^{\text{act}}(\mathbf{x}(t))|\right] > 1/2\cdot\left(2^{-(i+1)}\cdot n\right)$. Let us examine now the expected change in the Euclidean norm of $\mathbf{x}(t)$ during phase i.

$$\begin{aligned}
\mathbb{E}\left[||\mathbf{x}(\tau_{i+1})-\mathbf{x}(\tau_i)||_2^2\right] &\overset{\text{(i)}}{=} \gamma^2\cdot\mathbb{E}\left[\left|\left|\sum_{t=\tau_i}^{\tau_{i+1}-1} B(\mathbf{x}(t),t)\cdot\mathbf{g}(t)\right|\right|_2^2\right] \\
&\overset{\text{(ii)}}{=} \gamma^2\cdot\mathbb{E}\left[\sum_{t=\tau_i}^{\tau_{i+1}-1} Tr\left(B(\mathbf{x}(t),t)^2\right)\right] \\
&\geq \gamma^2 \sum_{t=\tau_i+b/(2\cdot\gamma^2)}^{\tau_{i+1}-1} \mathbb{E}\left[Tr\left(B(\mathbf{x}(t),t)^2\right)\right] \\
&\overset{\text{(iii)}}{\geq} \sum_{t=\tau_i+b/(2\cdot\gamma^2)}^{\tau_{i+1}-1} \left(\mathbb{E}\left[\left|\mathcal{C}^{\text{act}}(\mathbf{x}(t))\right|\right] - \mathbb{E}\left[|A(t)|\right]\right) \\
&\overset{\text{(iv)}}{>} \gamma^2 \sum_{t=\tau_i+b/(2\cdot\gamma^2)}^{\tau_{i+1}-1} \left(\frac{1}{4} - 2\cdot 2^{-\frac{1}{8}\frac{C^2\left(1-2^{-\frac{b}{2a}}\right)^2}{b}}\right)\cdot\left(2^{-i}\cdot n\right) \\
&= \frac{b}{2}\cdot\left(\frac{1}{4} - 2\cdot 2^{-\frac{1}{8}\frac{C^2\left(1-2^{-\frac{b}{2a}}\right)^2}{b}}\right)\cdot\left(2^{-i}\cdot n\right) \\
&\overset{\text{(v)}}{\geq} 4\cdot\left(2^{-i}\cdot n\right)
\end{aligned}$$

Equality (i) is by the definition of Algorithm 1. Note that equality (ii) follows from the fact that all the $\mathbf{g}(t)$s are independent random standard gaussian vectors. Specifically, it is easy to show that for any matrix A, vector $\mathbf{z}$ and random gaussian vector $\mathbf{g}$:

$$\mathbb{E}\left[||\mathbf{z}+A\mathbf{g}||_2^2\right] = ||\mathbf{z}||_2^2 + Tr\left(A\cdot A^T\right) \ .$$

Lemma 4.6 yields inequality (iii). Inequality (iv) is derived from Lemma 4.7 and the current case assumption that $\mathbb{E}\left[|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_{i+1}))|\right] > 1/2\cdot\left(2^{-(i+1)}\cdot n\right)$. Finally, inequality (v) follows from the conditions on the constants.

Note that $||\mathbf{x}(\tau_{i+1})-\mathbf{x}(\tau_i)||_2^2$ can *never* exceed $4\cdot|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_i))|$. This follows from Corollary 4.2, specifically that

$$E\left(B(\mathbf{x}(t),t)^2,\mathbf{x}(t)\right) \subseteq \left\{\mathbf{z}\in\mathbb{R}^n : z_i = x_i(t)\ \forall i\in U\setminus\mathcal{C}^{\text{act}}(\mathbf{x}(t))\right\},$$

which implies that all variables i that do not belong to $\mathcal{C}^{\text{act}}(\mathbf{x}(t))$ never change their value from iteration t onwards. On the other hand, all variables $i\in\mathcal{C}^{\text{act}}(\mathbf{x}(t))$ satisfy $x_i\in[-1,1]$.

Since phase $i-1$ was successful, we have that $|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_i))|\le 2^{-i}\cdot n$. Thus, $||\mathbf{x}(\tau_{i+1})-\mathbf{x}(\tau_i)||_2^2 \le 42^{-i}\cdot n$. This is a contradiction since we proved above that $\mathbb{E}\left[||\mathbf{x}(\tau_{i+1})-\mathbf{x}(\tau_i)||_2^2\right] > 4\cdot 2^{-i}\cdot n$. Thus, it cannot happen that $\mathbb{E}\left[|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_{i+1}))|\right] > 1/2\cdot\left(2^{-(i+1)}\cdot n\right)$ and we conclude the proof. ◀

We are now ready to prove the main result, Theorem 3.1.

Proof of Theorem 3.1. Let us calculate the probability that all phases Algorithm 1 makes are successful. First, denote by N the number of phases Algorithm 1 makes in case all phases are successful, and by T the total number of iterations in case all phases are successful. Definition 4.3 implies that $N = O(\log n)$, since $|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_i))|\le 2^{-i}\cdot n$. Second, Lemma 4.4 provides, conditioned on the success of the previous phase and by choosing $\gamma = b/n^2$, that the success probability of the current phase is at least $1/4$. Therefore, we can conclude that the probability that all phases of Algorithm 1 are successful is at least $(1/4)^N = 1/poly(n)$. Moreover, when all phases are successful the following two are implied:

1. Algorithm 1 never aborts (by Definition 4.3).
2. $\mathbf{x}(T)\in\mathcal{P}(T)$. Since $\mathcal{P}(T)\subseteq\mathcal{P}(\infty)$, we can conclude that for every $1\le j\le m$:

$$\left|\mathbf{1}_{S_j}\cdot\mathbf{x}(T)\right| \le C\cdot\sqrt{n\cdot\ln\left(\frac{2m}{n}\right)} \ .$$

Note that the rounding step of Algorithm 1 (step 10) might incur only an *additive* loss of 1 in the discrepancy. This concludes the proof. ◀

Choosing Parameters: It suffices to choose: $C=2^7$, $a=2^9$ and $b=2^6$ in order to satisfy all the required conditions.

5 Conclusion and Open Problems

We devise the first algorithm for Spencer's theorem that directly computes a coloring, without recursively computing partial colorings. This naturally leads to several interesting questions.

- Lovett-Meka [13] give a general condition (see Theorem 4 in [13]) when a partial coloring is present. Indeed, our algorithm can also be shown to give a *partial* coloring under these conditions. We defer the details to full version of the paper. A natural open question is whether our algorithm can also yield a partial coloring under the more general geometric condition as shown by Gluskin [12].

- Can this technique for directly producing full colorings be used to make progress on the Beck-Fiala [6] conjecture? As a first step, can this approach give an algorithmic form of Banaszczyk's result [2]?
- Although our algorithm does not directly produce partial colorings, our analysis involves multiple phases, which are somewhat analogous to partial colorings. Can the analysis be refined to avoid the notion of phases?

References

1 Noga Alon and Joel Spencer. *The Probabilistic Method*. John Wiley, 2000.
2 Wojciech Banaszczyk. Balancing vectors and Gaussian measures of n-dimensional convex bodies. *Random Structures & Algorithms*, 12(4):351–360, 1998.
3 N. Bansal and J. Spencer. Deterministic discrepancy minimization. *Algorithmica*, 67(4):451–471, 2013.
4 Nikhil Bansal. Constructive algorithms for discrepancy minimization. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 3–10. IEEE, 2010.
5 Nikhil Bansal, Moses Charikar, Ravishankar Krishnaswamy, and Shi Li. Better algorithms and hardness for broadcast scheduling via a discrepancy approach. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2014.
6 József Beck and Tibor Fiala. "Integer-making" theorems. *Discrete Applied Mathematics*, 3(1):1–8, 1981.
7 József Beck and Vera T. Sós. Discrepancy theory. In R. Graham and M. Grötschel and L. Lovász, editor, *Handbook of Combinatorics*, pages 1405–1446. Elsevier Science B.V., 1995.
8 Stephen P. Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
9 Bernard Chazelle. *The Discrepancy Method: Randomness and Complexity*. Cambridge University Press, 2000.
10 Friedrich Eisenbrand, Dömötör Pálvölgyi, and Thomas Rothvoß. Bin packing via discrepancy of permutations. *ACM Transactions on Algorithms (TALG)*, 9(3):24, 2013.
11 Apostolos Giannopoulos. On some vector balancing problems. *Studia Mathematica*, 122(3):225–234, 1997.
12 Efim Davydovich Gluskin. Extremal properties of orthogonal parallelepipeds and their applications to the geometry of Banach spaces. *Mathematics of the USSR-Sbornik*, 64(1):85, 1989.
13 Shachar Lovett and Raghu Meka. Constructive discrepancy minimization by walking on the edges. In *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 61–67. IEEE, 2012.
14 Jiří Matoušek. *Geometric discrepancy: An illustrated guide*, volume 18. Springer, 1999.
15 Thomas Rothvoß. Approximating bin packing within $O(\log \mathrm{OPT} \cdot \log\log \mathrm{OPT})$ bins. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 20–29. IEEE, 2013.
16 Thomas Rothvoß. Constructive discrepancy minimization for convex sets. *arXiv preprint arXiv:1404.0339*, 2014.
17 Joel Spencer. Six standard deviations suffice. *Transactions of the American Mathematical Society*, 289(2):679–706, 1985.
18 Aravind Srinivasan. Improving the discrepancy bound for sparse matrices: better approximations for sparse lattice approximation problems. In *Proceedings of the 8th annual ACM-SIAM Symposium on Discrete Algorithms*, pages 692–701. Society for Industrial and Applied Mathematics, 1997.

A Proof of Corollary 4.2

Proof. The proof follows the same outline but differs slightly from the proof of Theorem 4.1. First observe that the equality constraints for $\mathcal{P}^{\text{disc}}(t) \cap \{\mathbf{x} \in \mathbb{R}^n : x_i = \theta_i \ \forall i \in U \setminus \mathcal{C}^{\text{act}}(\theta)\}$ are $\mathbf{e}_i \cdot \mathbf{x} = \theta_i$ for each $i \in U \setminus \mathcal{C}^{\text{act}}(\theta)$. Now, we write the inequalities to make sure the constraints are orthogonal to the equality constraint. Observe that $\mathbf{1}_{S_j} \cdot \mathbf{x} \leq c_j(t)$ is equivalent to the constraint $\mathbf{1}_{S_j^{\text{act}}(\theta)} \cdot \mathbf{x} \leq c_j(t) - \mathbf{1}_{S \setminus S_j^{\text{act}}(\theta)} \cdot \theta$. Similarly, we have $-\mathbf{1}_{S_j} \cdot \mathbf{x} \leq c_j(t)$ is equivalent to the constraint $-\mathbf{1}_{S_j^{\text{act}}(\theta)} \cdot \mathbf{x} \leq c_j(t) + \mathbf{1}_{S \setminus S_j^{\text{act}}(\theta)} \cdot \theta$.

Thus, as in Theorem 4.1, we obtain that it is enough to show that the objective of the following semi-definite program is more than $\min_{J \subseteq \{1,\ldots,m\}:|J|=|\mathcal{C}^{\text{act}}(\theta)|} \left\{ \sum_{j \in J} \min\left\{1, d_j(\theta,t)^2\right\} \right\}$. Here the constraint $\Sigma \preceq I$ follows from the fact $E(\Sigma, \theta) \subseteq Ball(\theta, 1)$.

$$
\begin{aligned}
\max \quad & Tr(\Sigma) && \textit{(Primal-SDP2)} \\
s.t. \quad & \langle \mathbf{e}_i \mathbf{e}_i^T, \Sigma \rangle = 0 && \forall i \in U \setminus \mathcal{C}^{\text{act}}(\theta) \\
& \frac{\langle \mathbf{1}_{S_j^{\text{act}}(\theta)} \mathbf{1}_{S_j^{\text{act}}(\theta)}^T, \Sigma \rangle}{||\mathbf{1}_{S_j^{\text{act}}(\theta)}||_2^2} \leq \frac{c_j(t) - \mathbf{1}_{S \setminus S_j^{\text{act}}(\theta)} \cdot \theta - \mathbf{1}_{S^{\text{act}_j(\theta)}} \cdot \theta}{||\mathbf{1}_{S^{\text{act}_j(\theta)}}||_2^2} && \forall 1 \leq j \leq m \\
& \frac{\langle \mathbf{1}_{S^{\text{act}_j(\theta)}} \mathbf{1}_{S^{\text{act}_j(\theta)}}^T, \Sigma \rangle}{||\mathbf{1}_{S^{\text{act}_j(\theta)}}||_2^2} \leq \frac{c_j(t) + \mathbf{1}_{S \setminus S_j^{\text{act}}(\theta)} \cdot \theta + \mathbf{1}_{S_j^{\text{act}}(\theta)} \cdot \theta}{||\mathbf{1}_{S_j^{\text{act}}(\theta)}||_2^2} && \forall 1 \leq j \leq m \\
& \Sigma \preceq I \\
& \Sigma \succeq 0
\end{aligned}
$$

Simplifying, we obtain that the SDP is equivalent to

$$
\begin{aligned}
\max \quad & Tr(\Sigma) && \textit{(Primal-SDP2)} \\
s.t. \quad & \langle \mathbf{e}_i \mathbf{e}_i^T, \Sigma \rangle = 0 && \forall i \in U \setminus \mathcal{C}^{\text{act}}(\theta) \\
& \left(||\mathbf{1}_{S_j^{\text{act}}(\theta)}||_2^2 \right)^{-1} \langle \mathbf{1}_{S_j^{\text{act}}(\theta)} \mathbf{1}_{S_j^{\text{act}}(\theta)}^T, \Sigma \rangle \leq d_j(\theta,t)^2 && \forall 1 \leq j \leq m \\
& \Sigma \preceq I \\
& \Sigma \succeq 0
\end{aligned}
$$

Consider the dual of *Primal-SDP2*:

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{m} \lambda_j d_j(\theta)^2 + Tr(V) && \textit{(Dual-SDP2)} \\
s.t. \quad & \sum_{i \in U \setminus \mathcal{C}^{\text{act}}(\theta)} \mu_i \left(\mathbf{e}_i \mathbf{e}_i^T \right) + \sum_{j=1}^{m} \lambda_j \frac{1}{|S_j^{\text{act}}(\theta)|} \left(\mathbf{1}_{S_j^{\text{act}}(\theta)} \mathbf{1}_{S_j^{\text{act}}(\theta)}^T \right) \succeq I - V \\
& \lambda_j \geq 0 && \forall 1 \leq j \leq m \\
& V \succeq 0
\end{aligned}
$$

By renaming the constraints assume, without loss of generality, that $d_1(\theta,t) \leq \ldots \leq d_m(\theta,t)$ and r be the maximum value such that $d_r(\theta,t) \leq 1$. Assume that $r \leq |\mathcal{C}^{\text{act}}(\theta)|$, else the proof is identical to proof of Theorem 4.1. We will show that the dual objective value for any feasible dual solution (λ, μ, V) is at least $\sum_{j=1}^{r} d_j(\theta,t)^2 + |\mathcal{C}^{\text{act}}(\theta)| - r$ which will prove the desired result.

For any $0 \leq t \leq |\mathcal{C}^{\text{act}}(\theta)|$, consider the subspace S_t that is orthogonal to the vectors $\{\mathbf{a}_i\}_{i \in U \setminus \mathcal{C}^{\text{act}}(\theta)}$ and the vectors $\{\mathbf{v}_j\}_{j=1}^{t}$. Note that the dimension of S_t is at least $|\mathcal{C}^{\text{act}}(\theta)| - t$. Denote by B_t the matrix whose columns form an orthonormal basis of S_t. Taking the inner product of the dual constraint with $B_t B_t^T$, we obtain that:

$$B_t B_t^T \cdot \left(\sum_{i \in U \setminus \mathcal{C}^{\text{act}}(\theta)} \mu_i \left(\mathbf{e}_i \mathbf{e}_i^T \right) + \sum_{j=1}^{m} \lambda_j \frac{1}{|S_j^{\text{act}}(\theta)|} \left(\mathbf{1}_{S_j^{\text{act}}(\theta)} \mathbf{1}_{S_j^{\text{act}}(\theta)}^T \right) \right) \succeq B_t B_t^T \cdot I - B_t B_t^T \cdot V \ . \quad (5)$$

As in proof of Theorem 4.1, we conclude that l.h.s. is upper bounded by $\sum_{j=t+1}^{m} \lambda_j$. Let us focus now on the r.h.s. of (5):

$$\left(B_t B_t^T \right) \cdot I - B_t B_t^T \cdot V \geq Tr(B_t B_t^T) - Tr(V) \geq |\mathcal{C}^{\text{act}}(\theta)| - t - Tr(V) \ .$$

where we use the fact that $B_t B_t^T \cdot V \leq I \cdot V = Tr(V)$. Thus, we obtain that for each $1 \leq t \leq m$:

$$\sum_{j=t+1}^{m} \lambda_j \geq |\mathcal{C}^{\text{act}}(\theta)| - t - Tr(V) \ .$$

Consider the following linear program with variables λ_j for each $1 \leq j \leq m$ and variable $Tr(V)$ which is a relaxation of (Primal-SDP). Thus it enough to lower bound the value of optimum solution to this linear program.

$$\begin{aligned} \min \quad & \sum_{j=1}^{m} \lambda_j d_j(\theta)^2 + Tr(V) & \\ s.t. \quad & \sum_{j=t+1}^{m} \lambda_j \geq |\mathcal{C}^{\text{act}}(\theta)| - t - Tr(V) \ . & 0 \leq t \leq |\mathcal{C}^{\text{act}}(\theta)| \\ & \lambda_j \geq 0 & \forall 1 \leq j \leq m \end{aligned}$$

Since $d_1(\theta) \leq \ldots \leq d_m(\theta)$ and all the λ_js are non-negative and $d_j(\theta) > 1$ if $j > r$, the optimal solution to the above linear program is $\lambda_j = 1$ for each $1 \leq j \leq r$ and $\lambda_j = 0$ for each $j > r$ and $Tr(V) = |\mathcal{C}^{\text{act}}(\theta)| - r$. Thus we can conclude that $\sum_{j=1}^{m} \lambda_j d_j(\theta)^2 + Tr(V) \geq \sum_{j=1}^{r} d_j(\theta)^2 + |\mathcal{C}^{\text{act}}(\theta)| - r \geq \sum_{j=1}^{|\mathcal{C}^{\text{act}}(\theta)|} \min\{1, d_j(\theta)^2\}$ as claimed. ◀

B Proof of Lemma 4.5

Proof.

$$\begin{aligned} & \Pr\left[\mathbf{x}(t+1) \notin \mathcal{P}(t) | \mathbf{x}(t) \in \mathcal{P}(t)\right] \\ \overset{\text{(i)}}{=} & \Pr\left[\mathbf{x}(t) + \gamma \cdot B(\mathbf{x}(t), t) \cdot \mathbf{g}(t) \notin \mathcal{P}(t) | \mathbf{x}(t) \in \mathcal{P}(t)\right] \\ \overset{\text{(ii)}}{\leq} & \Pr\left[\mathbf{x}(t) + \gamma \cdot B(\mathbf{x}(t), t) \cdot \mathbf{g}(t) \notin E\left(B(\mathbf{x}(t), t)^2, \mathbf{x}(t)\right) | \mathbf{x}(t) \in \mathcal{P}(t)\right] + \\ & \Pr\left[||\mathbf{x}(t) + \gamma \cdot B(\mathbf{x}(t), t) \cdot \mathbf{g}(t)||_\infty > 1 | \mathbf{x}(t) \in \mathcal{P}(t)\right] \\ = & \Pr\left[\gamma \cdot B(\mathbf{x}(t), t) \cdot \mathbf{g}(t) \notin E\left(B(\mathbf{x}(t), t)^2, 0\right) | \mathbf{x}(t) \in \mathcal{P}(t)\right] + \\ & \Pr\left[||\mathbf{x}(t) + \gamma \cdot B(\mathbf{x}(t), t) \cdot \mathbf{g}(t)||_\infty > 1 | \mathbf{x}(t) \in \mathcal{P}(t)\right] \end{aligned}$$

$$
\begin{aligned}
&= \Pr\left[||\mathbf{g}(t)||_2^2 > 1/\gamma^2\right] + \\
&\quad \Pr\left[\exists i \in \mathcal{C}^{\text{act}}(\mathbf{x}(t)) \; s.t. \; |x_i(t) + \gamma \cdot (B(\mathbf{x}(t),t) \cdot \mathbf{g}(t))_i| > 1 | \mathbf{x}(t) \in \mathcal{P}(t)\right] \\
&\overset{\text{(iii)}}{\leq} \Pr\left[||\mathbf{g}(t)||_2^2 > 1/\gamma^2\right] + \\
&\quad \Pr\left[\exists i \in \mathcal{C}^{\text{act}}(\mathbf{x}(t)) \; s.t. \; \gamma \cdot |(B(\mathbf{x}(t),t) \cdot \mathbf{g}(t))_i| > 1/n | \mathbf{x}(t) \in \mathcal{P}(t)\right] \\
&\overset{\text{(iv)}}{\leq} (2n+1) \cdot \left(1 - \Phi\left(\frac{1}{\gamma \cdot n}\right)\right)
\end{aligned}
$$

Equality (i) is from the definition of Algorithm 1. Inequality (ii) is derived from the definition of $\mathcal{P}(t)$ and Corollary 4.2 since: $E\left(B(\mathbf{x}(t),t)^2, \mathbf{x}(t)\right) \subseteq \mathcal{P}^{\text{disc}}(t)$. Inequality (iii) is derived from the fact that $i \in \mathcal{C}^{\text{act}}(\mathbf{x}(t))$ implies that $|x_i(t)| < 1 - 1/n$. Finally, inequality (iv) is true since $|\mathcal{C}^{\text{act}}(\mathbf{x}(t))| \leq n$ and since Corollary 4.2 implies that $E\left(B(\mathbf{x}(t),t)^2, \mathbf{x}(t)\right) \subseteq \text{Ball}(\mathbf{x}(t), 1)$. The lemma now follows since $\mathcal{P}(t) \subseteq \mathcal{P}(t+1)$. Note that the exact same proof holds also when conditioning that phase $i-1$ is successful. ◀

C Proof of Lemma 4.7

Proof. For simplicity, let us denote $t = \tau_i + s$ where $b/(2\gamma^2) \leq s < b/\gamma^2$.

$$
\begin{aligned}
\Pr\left[d_j\left(\mathbf{x}(t),t\right) \leq 1\right] &\overset{\text{(i)}}{=} \Pr\left[\frac{c_j(t) - \left|\mathbf{1}_{S_j} \cdot \mathbf{x}(t)\right|}{||\mathbf{1}_{S_j^{\text{act}}\mathbf{x}(t)}||_2} \leq 1\right] \\
&= \Pr\left[c_j(t) - ||\mathbf{1}_{S_j^{\text{act}}(\mathbf{x}(t))}||_2 \leq \left|\mathbf{1}_{S_j} \cdot (\mathbf{x}(t) - \mathbf{x}(\tau_i)) + \mathbf{1}_{S_j} \cdot \mathbf{x}(\tau_i)\right|\right] \\
&\overset{\text{(ii)}}{\leq} \Pr\left[\left|\mathbf{1}_{S_j} \cdot (\mathbf{x}(t) - \mathbf{x}(\tau_i))\right| \geq c_j(t) - c_j(\tau_i) - ||\mathbf{1}_{S_j^{\text{act}}(\mathbf{x}(t))}||_2\right] . \qquad (6)
\end{aligned}
$$

Equality (i) is by the definition of $d_j(\mathbf{x}(t),t)$. Inequality (ii) follows from the fact that phase $i-1$ is successful, and in particular Algorithm 1 did not abort (implying that $|\mathbf{x}(\tau_i)| \leq c_j(\tau_i)$). Let us now lower bound the r.h.s. of the event in (6). First,

$$
\begin{aligned}
c_j(t) - c_j(\tau_i) &= C \cdot \sqrt{n \cdot \ln\left(2m/n\right)} \cdot 2^{-\frac{b}{a} \cdot i} \cdot \left(1 - 2^{-\frac{s \cdot \gamma^2}{a}}\right) \\
&\overset{\text{(iii)}}{\geq} C \cdot \sqrt{n \cdot \ln\left(2m/n\right)} \cdot 2^{-\frac{b}{a} \cdot i} \cdot \left(1 - 2^{-\frac{b}{2a}}\right) . \qquad (7)
\end{aligned}
$$

Inequality (iii) is derived from the fact that we consider only iterations t in the second half of phase i, i. e., $b/(2\gamma^2) \leq s$. Second,

$$
||\mathbf{1}_{S_j^{\text{act}}(\mathbf{x}(t))}||_2 \leq \sqrt{\left|S_j^{\text{act}}(\mathbf{x}(t))\right|} \leq \sqrt{|\mathcal{C}^{\text{act}}(\mathbf{x}(t))|} \overset{\text{(iv)}}{\leq} \sqrt{|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_i))|} \overset{\text{(v)}}{\leq} 2^{-i/2} \cdot \sqrt{n} . \qquad (8)
$$

Note that $|\mathcal{C}^{\text{act}}(\mathbf{x}(t))|$ is a monotone non-increasing function in t, and thus inequality (iv) follows. Since we assumed that phase $i-1$ is successful, i. e., $|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_i))| \leq 2^{-i} \cdot n$ (see Definition 4.3), inequality (v) is true. Plugging (7) and (8) into the r.h.s. of the event in (6) yields:

$$
\begin{aligned}
c_j(t) - c_j(\tau_i) - ||\mathbf{1}_{S_j^{\text{act}}(\mathbf{x}(t))}||_2 &\geq C \cdot \sqrt{n \cdot \ln\left(2m/n\right)} \cdot 2^{-\frac{b}{a} \cdot i} \cdot \left(1 - 2^{-\frac{b}{2a}}\right) - 2^{-i/2} \cdot \sqrt{n} \\
&\overset{\text{(vi)}}{\geq} \sqrt{n} \cdot 2^{-\frac{b}{a} \cdot i} \cdot \left(C \cdot \sqrt{\ln\left(2m/n\right)} \cdot \left(1 - 2^{-\frac{b}{2a}}\right) - 1\right) \\
&\overset{\text{(vii)}}{\geq} \sqrt{n} \cdot 2^{-\frac{b}{a} \cdot i} \cdot \frac{C}{2} \cdot \sqrt{\ln\left(2m/n\right)} \cdot \left(1 - 2^{-\frac{b}{2a}}\right) \qquad (9)
\end{aligned}
$$

Inequalities (vi) and (vii) are both derived from the conditions we imposed on the constants. Specifically, inequality (vi) follows from the condition that $a \geq 8b$, whereas inequality (vii) follows from the condition that $C \cdot \left(1 - 2^{-b/(2a)}\right) \geq \sqrt{32b} \geq 4$ (since $b \geq 64$). Next, consider the l.h.s. of the event in (6). Note that by the definition of Algorithm 1:

$$\mathbf{x}(t) - \mathbf{x}(\tau_i) = \gamma \sum_{r=\tau_i}^{\tau_i+s} B(\mathbf{x}(r), r) \cdot \mathbf{g}(r) \ .$$

It can be verified that given the random choices of the algorithm in the first $r-1$ iterations, for any $\tau_i \leq r \leq \tau_i + s$, the random variable $\gamma \mathbf{1}_{S_j} \cdot (B(\mathbf{x}(r), r) \cdot \mathbf{g}(r))$ is a normal random variable with mean 0 and standard deviation σ, where:

$$\sigma^2 = \gamma^2 \mathbf{1}_{S_j}^T B(\mathbf{x}(r), r)^2 \mathbf{1}_{S_j} \overset{\text{(viii)}}{=} \gamma^2 \mathbf{1}_{S_j^{\text{act}}(\mathbf{x}(r))}^T B(\mathbf{x}(r), r)^2 \mathbf{1}_{S_j^{\text{act}}(\mathbf{x}(r))} \tag{10}$$

$$\overset{\text{(ix)}}{\leq} \gamma^2 \left|S_j^{\text{act}}(\mathbf{x}(r))\right| \overset{\text{(x)}}{\leq} \gamma^2 \cdot 2^{-i} n \ . \tag{11}$$

Equality (viii) is derived from the property that

$$E(B(\mathbf{x}(r), r)^2, \mathbf{x}(r)) \subseteq \left\{\mathbf{z} \in \mathbb{R}^n : z_i = x_i(r) \ \forall i \in U \setminus \mathcal{C}^{\text{act}}(\mathbf{x}(r))\right\} \ ,$$

as guaranteed by Corollary 4.2. Note that inequality (ix) follows again from Corollary 4.2, specifically that $B(\mathbf{x}(r), r)^2 \preceq I$ (or equivalently that $E(B(\mathbf{x}(r), r)^2, \mathbf{x}(r)) \subseteq \text{Ball}(\mathbf{x}(r), 1)$). Finally, recall that $|\mathcal{C}^{\text{act}}(\mathbf{x}(r))|$ is a monotone non-increasing function in r. Therefore, inequality (x) is true since we assume that phase $i-1$ was successful, i. e., $|\mathcal{C}^{\text{act}}(\mathbf{x}(\tau_i))| \leq 2^{-i} \cdot n$ (see Definition 4.3). Applying the concentration bound of [4] for (6) results in:

$$\begin{aligned}
\Pr\left[d_j(\mathbf{x}(t), t) \leq 1\right] &\leq \Pr\left[\left|\mathbf{1}_{S_j} \cdot (\mathbf{x}(t) - \mathbf{x}(\tau_i))\right| \geq c_j(t) - c_j(\tau_i) - ||\mathbf{1}_{S_j^{\text{act}}(\mathbf{x}(t))}||_2\right] \\
&\overset{\text{(xi)}}{\leq} 2 \cdot \exp\left(-\frac{1}{2} \cdot \frac{n \cdot \frac{1}{4} \cdot C^2 \cdot 2^{-\frac{2b}{a} \cdot i} \cdot \left(1 - 2^{-\frac{b}{2a}}\right)^2 \cdot \ln\left(\frac{2m}{n}\right)}{\left(\gamma \cdot 2^{-i/2} \cdot \sqrt{n}\right)^2 \cdot s}\right) \\
&\overset{\text{(xii)}}{\leq} 2 \cdot \exp\left(-\frac{1}{8} \cdot \frac{C^2 \cdot \left(1 - 2^{-\frac{b}{2a}}\right)^2}{b} \cdot 2^{\left(1 - \frac{2b}{a}\right) \cdot i} \cdot \ln\left(\frac{2m}{n}\right)\right) \\
&= 2 \cdot \left(\frac{n}{2m}\right)^{\frac{1}{8} \cdot \frac{C^2 \cdot \left(1 - 2^{-\frac{b}{2a}}\right)^2}{b} \cdot 2^{\left(1 - \frac{2b}{a}\right) \cdot i}} \\
&\overset{\text{(xiii)}}{\leq} 2 \cdot \left(\frac{n}{2m}\right)^{\frac{1}{8} \cdot \frac{C^2 \cdot \left(1 - 2^{-\frac{b}{2a}}\right)^2}{b}} \cdot 2^{-2^{\left(1 - \frac{2b}{a}\right) \cdot i}} \\
&\overset{\text{(xiv)}}{\leq} 2 \cdot \left(\frac{n}{2m}\right)^{\frac{1}{8} \cdot \frac{C^2 \cdot \left(1 - 2^{-\frac{b}{2a}}\right)^2}{b}} \cdot 2^{-i} \\
&\overset{\text{(xv)}}{\leq} 2 \cdot 2^{-\frac{1}{8} \cdot \frac{C^2 \cdot \left(1 - 2^{-\frac{b}{2a}}\right)^2}{b}} \cdot \frac{n}{m} \cdot 2^{-i}
\end{aligned}$$

Inequality (xi) is obtained by plugging into the tail bound of [4] inequalities (11) and (9) for any $\tau_i \leq r \leq \tau_i + s$. Inequality (xii) follows since $s \leq b/\gamma^2$. Inequality (xiii) is derived from the conditions on the constants, specifically that $C \cdot \left(1 - 2^{-b/(2a)}\right) \geq \sqrt{32b}$, and the fact

that $m \geq n$ (and hence $n/(2m) \leq 1/2$). Inequality (xiv) is derived again from the conditions on the constants, specifically that $a \geq 8b$, which implies that $2^{\left(1-\frac{2b}{a}\right)\cdot i} \geq i$ for every $i \geq 0$. Inequality (xv), similarly to (xiii), is derived from $C \cdot \left(1 - 2^{-b/(2a)}\right) \geq \sqrt{32b}$. Linearity of expectation concludes the proof since $|\mathcal{S}| = m$. ◀

Universal Factor Graphs for Every NP-Hard Boolean CSP

Shlomo Jozeph

Weizmann Institute of Science
Rehovot, Israel
shlomo.jozeph@weizmann.ac.il

Abstract

An instance of a Boolean constraint satisfaction problem can be divided into two parts. One part, that we refer to as the *factor graph* of the instance, specifies for each clause the set of variables that are associated with the clause. The other part, specifies for each of the given clauses what is the constraint that is evaluated on the respective variables. Depending on the allowed choices of constraints, it is known that Boolean constraint satisfaction problems fall into one of two classes, being either NP-hard or in P.

This paper shows that every NP-hard Boolean constraint satisfaction problem (except for an easy to characterize set of natural exceptions) has a universal factor graph. That is, for every NP-hard Boolean constraint satisfaction problem, there is a family of at most one factor graph of each size, such that the problem, restricted to instances that have a factor graph from this family, cannot be solved in polynomial time unless $\mathsf{NP} \subset \mathsf{P/poly}$. Moreover, we extend this classification to one that establishes hardness of approximation.

1998 ACM Subject Classification F. Theory of Computation

Keywords and phrases Hardness of Approximation, Hardness with Preprocessing

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.274

1 Introduction

A Boolean constraint satisfaction problem (CSP) C is defined by a set of allowable constraints. An instance of C consists of m constraints (from the set of allowable constraints) over n variables. The goal is to assign Boolean values to the variables in order to maximize the number of satisfied constraints. [9, 4] showed CSPs can be divided into three types:

- CSPs for which there is a constant $\varsigma < 1$ such that it is NP-hard to decide if there is a assignment satisfying all constraints or every assignment satisfies at most ς-fraction of the constraints.
- CSPs for which there is a polynomial time algorithm finding an assignment satisfying all constraints if it exists, but there are constants $\varsigma < \kappa < 1$ such that it is NP-hard to decide if there is an assignment that satisfies at least κ-fraction of the constraints or every assignment satisfies at most ς-fraction of the constraints.
- CSPs for which there is a polynomial time algorithm finding an assignment satisfying the maximum number of constraints.

An instance of a CSP can be divided into two parts; the graph structure connecting the constraints and the variables, and the choice of the specific constraints from the set of allowable constraints. We call the structure of a CSP a factor graph. If the factor graph is known, what can be said about the hardness of solving or approximating a CSP instance?

This question is formalized in the following way: given a CSP C, a family $\mathcal{F}$ of factor graphs for C (at most one for each size) is called a universal factor graph (UFG) for C if it

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 274–283

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

is NP-hard to decide if an instance of C with a factor graph from $\mathcal{F}$ is satisfiable. If it is NP-hard to decide if the instance is at least κ-satisfiable or at most ς-satisfiable, $\mathcal{F}$ is called a (κ, ς)-universal factor graph for C.

The existence of a UFG for C implies that at least some of the hardness of *solving* instances of C does not come from the structure of the instances. The existence of (κ, ς)-UFG for C implies that at least part of the hardness of *maximizing* the number of satisfied constraints does not come from the structure of instances. On the other hand, if no UFG for C exists and C is NP-hard to solve, the hardness of solving C comes from the structure of the instances.

In this paper, we continue the work of [6], that introduced the notion of UFG and showed UFGs for several CSPs. We show similar results to those of [9, 5]. Specifically, every CSP that is NP-hard has a (κ, ς)-UFG, for some $0 < \varsigma < \kappa \leq 1$ that depend on the CSP (unless the existence of such a UFG trivially implies that NP $\subset$ P/poly). Additionally, if it is NP-hard to decide if a CSP instance has an assignment satisfying all clauses, then this CSP has a $(1, \varsigma)$-UFG.

1.1 Definitions

The definitions used here are similar to those of [5].

▶ **Definition 1.** A *constraint* is a function $f : \{0,1\}^k \to \{0,1\}$. k is the *arity* of the constraint. The constraint is *satisfied* by an assignment x if $f(x) = 1$.

We define the following constraints: $\text{FALSE}_k : \{0,1\}^k \to \{0\}$, $\text{TRUE}_k : \{0,1\}^k \to \{1\}$. The arity will be known from the context, so the subscript will usually be omitted. The constraints ID and NOT have arity 1 and return the input and its negation, respectively. We call a constraint that is not FALSE a *satisfiable constraint*.

▶ **Definition 2.** A *constraint set* is a finite set of constraints, all with the same arity. The arity of a constraint set is the arity of the constraints it contains.

For example, the constraint set corresponding to 2LIN contains two constraints, f_1 and f_2, with $f_1(x, y) = x \oplus y$ and $f_2(x, y) = x \oplus y \oplus 1$. The constraint set corresponding to 3SAT contains eight constraints, which are all the possible 3CNF clauses on three variables.

▶ **Definition 3.** A *constraint application* is an ordered set $\langle f, i_1, \cdots, i_k \rangle$, where f is a constraint of arity k, and each i_j is a natural number indicating the name of the variable.

The same variable name may appear several times in a constraint application.

▶ **Definition 4.** Given a constraint set S, a *formula over S with n variables* is a (multi)set P containing constraint applications. For each $c \in P$, $c = \langle f, i_1, \cdots, i_k \rangle$, where k is the arity of S, $f \in S$, and $i \in [n]$. Usually, S and n will be implied from the context.

An *assignment* $x \in \{0,1\}^n$ satisfies a constraint application $\langle f, i_1, \cdots, i_k \rangle$ if $f(x_{i_1}, \cdots, x_{i_k}) = 1$. A formula is α-satisfied by an assignment x if an α-fraction of the constraints in the formula are satisfied. A formula is α-satisfiable if there is an assignment that α-satisfies it. In the case of $\alpha = 1$ we may omit α.

A formula may contain several copies of the same constraint application. This is modeled as giving each constraint application an integer weight, that is, if a constraint application has weight w, there are w copies of it in the formula.

▶ **Definition 5.** A constraint set S will be called NP-*hard* if it is NP-hard to decide satisfiability for formulas over S. A constraint set S will be called APX-*hard* if there are constants $0 \leq \varsigma < \kappa \leq 1$ such that it is NP-hard to decide whether a formula over S is at least κ-satisfiable or at most ς-satisfiable. The approximation hardness of S is st least ς/κ.

▶ **Definition 6.** Given a formula P over n variables, its corresponding *factor graph* is a bipartite graph $G = (V \biguplus C, E)$. The edges of each vertex in C are ordered. $|V| = n$, and each vertex in V is associated with a variable. $|C| = |P|$, and each vertex of C is associated with an element of P. For each $c = \langle f, i_1, \cdots, i_k \rangle$ and $j \in [k]$, there is an edge (c, i_j), the j'th edge of c.

The size of a factor graph $G = (V \biguplus C, E)$ is $|V| + |C| + |E|$.

Since a variable name may appear in a constraint application several times, the factor graph may have parallel edges.

▶ **Definition 7.** Given an APX-hard constraint set S, a (κ, ς)-*universal factor graph (UFG) over* S is a family of factor graphs $G = \{G_n\}$, at most one graph of each size, such that deciding if a formula over S that has a factor graph from the family is at least κ-satisfiable or at most ς-satisfiable is NP-hard. If $\kappa = 1$ we say that the UFG has perfect completeness.

1.2 Our Results

▶ **Theorem 8.** *Assume that* NP $\not\subset$ P/poly.

1. *If a constraint set S is* NP*-hard and has at least two satisfiable constraints, then S has a $(1, \varsigma)$-universal factor graph with a constant $\varsigma < 1$ that depends only on S.*
2. *If a constraint set S is* APX*-hard and has at least two constraints, then S has a (κ, ς)-universal factor graph with constants $0 < \varsigma < \kappa < 1$ that depends only on S.*

There are CSPs that have only one constraint in their corresponding constraint set. For example, ONE_IN_THREE and 2XOR (see [5] for proofs that ONE_IN_THREE is NP-hard and 2XOR is APX-hard). If a constraint set contains only one constraint, then each instance is defined completely by its factor graph. A UFG for such a CSP will contain only one instance for each size. Thus, having a UFG for a CSP that has only one constraint implies that NP $\subset$ P/poly, as the answer for each instance can be given by the advice.

We show that if an NP-hard constraint set has at least two constraints (excluding the constraint FALSE) then it has a UFG with perfect completeness and constant approximation hardness. Similarly, every constraint set that is APX-hard with at least two constraints (in this case, one of them can be FALSE) has a UFG with constant approximation hardness. Note that the constants depend on the constraint set.

1.3 Related Work

The term universal factor graph was introduced in [6]. A $(1, 77/80)$-UFG for 3SAT was constructed, and was used used to construct UFGs for several other CSPs, but the existence of a UFG for all CSPs remained an open question.

The notion of hardness of preprocessing (see [2, 7], for example) considers problems where the input can naturally be partitioned into two parts, and one of them is known in advance. For example, the problem of nearest codeword: given an input (A, s) where A is a generating matrix for a linear codeword, the goal is to find the closest codeword to s. In certain scenarios, it is natural to assume that A is known in advance and will be used for many instances. In such cases, investing time in preprocessing A may be beneficial. However,

it turns out that even preprocessing A does not help to find the nearest codeword to s in polynomial time (but preprocessing may improve the approximation ratio). Similarly, the existence of UFG for a CSP C shows that preprocessing the structure of an instance of C does not give rise to a polynomial time solution to instances of C. See [6] for a more in-depth discussion about the connection between UFGs and hardness with preprocessing.

Classifying Boolean CSPs into NP-hard to solve and solvable in P was done by Schaefer [9]. Schaefer has shown that there are only NP-hard CSPs and CSPs that are solvable in P. Creignou [4] have later shown that all Boolean CSPs are either NP-hard to approximate or can be maximized in P. Classifying non-Boolean CSPs is still an open question and an area of active research, see [3] for a survey on the subject.

Another type of classification is that of approximation resistance. Loosely speaking, a problem is approximation resistant if it is hard to approximate it better than the approximation ratio of a random assignment. To date, there is no complete classification of approximation resistant CSPs. However, there is a complete classification of related notions of strong inapproximability [8] and usefulness [1] assuming the Unique Games Conjecture.

2 Proofs

2.1 Preliminaries

▶ **Definition 9.** Let f be a constraint of arity k, and g be a formula over S with $n \geq k$ variables. The variables of g are $x_1, \cdots, x_k$ and $y_{k+1}, \cdots, y_n$. The formula g is called an $f\left(x_1, \cdots, x_k\right)^{c,s}$*-formula over* S if the following conditions hold:

1. Every assignment to the variables cannot satisfy more than c of the constraint applications in g.
2. (Completeness) For every assignment to the x variables that satisfies f, there must be an assignment to the y variables that satisfies c of the constraint applications in g.
3. (Soundness) Given an assignment to the x variables that does not satisfy f and any assignment to the y variables, at most s of the constraint applications in g are satisfied.
4. For every assignment to the x variables that does not satisfy f, there must be an assignment to the y variables that satisfies s of the constraint applications in g.

The x variables are called the *primary variables* and the y variables are called the *auxiliary variables*.

▶ **Definition 10.** An $f\left(x_1, \cdots, x_k\right)$-formula over S is similar to a $f\left(x_1, \cdots, x_k\right)^{c,s}$-formula over S in the special case that c is the number of constraints in g, $s < c$ and moreover, condition 4 need not hold.

For example, the formula over 2SAT $\{x_1 \vee y_3, x_2 \vee \overline{y}_3\}$ is a 2SAT $\left(x_1, x_2\right)^{2,1}$-formula over 2SAT. x_1 and x_2 are the primary variables, and y_3 is the auxiliary variable. If $x_1 \vee x_2$ is true, y_3 can be set in a way to satisfy both clauses (true if x_1 is false, false if x_2 is false). Otherwise, exactly one clause will be satisfied.

The basic building blocks of a UFG for a general CSP are *templates*. Templates can be thought of as a set of placeholders for constraints, and, depending on the constraints chosen to be used, the template is instantiated to be one of several specific formulas.

Consider a set of t formulas $\{g_i\}_{i=1}^{t}$, where g_i is an $f_i\left(x_1, \cdots, x_k\right)^{c,s}$-formula over S. Furthermore, $g_i = \left\{\left\langle g_i^j, r_1^j, \cdots, r_m^j \right\rangle\right\}_{j=1}^{q}$. That is, each g_i has the same number of constraints, and the j'th constraint application of g_i depends on exactly the same variables as the j'th constraint application of $g_{i'}$, in the same order. This means that all formulas $\{g_i\}$ have the

same factor graph G, the same completeness and soundness, and they all depend on the same set of primary and auxiliary variables, in the same order. We call the factor graph G an $(f_1, \cdots, f_t)^{c,s}$-*template over* S.

If the same conditions hold, except that each g_i is an $f_i(x_1, \cdots, x_k)$-formula over S instead of an $f_i(x_1, \cdots, x_k)^{c,s}$-formula over S, we call the factor graph G an $(f_1, \cdots, f_t)$-*template over* S. In this case, the formulas $\{g_i\}$ may not have the same soundness, but they still have the same completeness, and they all depend on the same set of primary and auxiliary variables, in the same order.

In order to simplify the proofs, we will expand the definition of formulas and allow to force some variables to have a certain constant value (0 or 1). Later, we will show how to get rid of this use of constants.

▶ **Lemma 11.** *If a constraint set S has two different satisfiable constraints, and we can use constants in place of variables, then at least one of the following templates over S exists:* $(\mathrm{ID}, \mathrm{NOT})^{1,0}$, $(\mathrm{NOT}, \mathrm{TRUE})^{1,0}$, $(\mathrm{ID}, \mathrm{TRUE})^{1,0}$.

Before proving the lemma, we show a simple example: Suppose S contains the constraints 2XOR and 2SAT. There is an assignment satisfying 2SAT but not 2XOR, $\langle 1, 1\rangle$. There is an assignment satisfying 2XOR, for example $\langle 0, 1\rangle$. Both assignments have 1 in the second bit. $\mathrm{2XOR}(x, 1)$ is $\mathrm{NOT}(x)$ and $\mathrm{2SAT}(x, 1)$ is $\mathrm{TRUE}(x)$, so this is a $(\mathrm{NOT}, \mathrm{TRUE})^{1,0}$-template. As another example, suppose that S contains 2EQ (the complement of 2XOR) and 2XOR. $\mathrm{2EQ}(x, 1)$ is $\mathrm{ID}(x)$ and $\mathrm{2XOR}(x, 1)$ is $\mathrm{NOT}(x)$, so this is an $(\mathrm{ID}, \mathrm{NOT})^{1,0}$-template.

Proof. Let c_1, c_2 be two different satisfiable constraints in S. Let a, b be two assignments, b satisfying c_2 but not c_1, and a satisfying c_1 (if the set of the assignments satisfying c_2 is contained in the set of assignments satisfying c_1, switch between them). We will create two formulas with the same factor graph that will show the existence of one of the templates. The formula g_1 only contains c_1, and the formula g_2 only contains c_2. Both formulas have one primary variable.

In the indices where a and b are 0, we put the constant 0. In the indices where a and b are 1, we put the constant 1. In the indices where a is 1 and b is 0, we put the (new) variable t_{10}. In the indices where a is 0 and b is 1, we put the (new) variable t_{01}. Since $a \neq b$, the variable t_{01} or the variable t_{10} must appear.

1. If t_{10} does not appear, the primary variable is t_{01}. Since all the other variables are constants, the only assignment where t_{01} is 0 is a, and b is the only assignment where t_{01} is 1. $c_1(b) = 0$ but $c_2(b) = 1$. $c_1(a) = 1$, but we don't know what $c_2(a)$ is. g_1 is a $\mathrm{NOT}^{1,0}$-formula and g_2 is either an $\mathrm{ID}^{1,0}$-formula or a $\mathrm{TRUE}^{1,0}$-formula. Thus, we get either a $(\mathrm{NOT}, \mathrm{ID})^{1,0}$-template (which is the same as an $(\mathrm{ID}, \mathrm{NOT})^{1,0}$-template) or a $(\mathrm{NOT}, \mathrm{TRUE})^{1,0}$-template, depending on whether $c_2(a)$ is false or true, respectively.
2. If t_{01} does not appear, the primary variable is t_{10}. Since all the other variables are constants, the only assignment where t_{10} is 1 is a, and b is the only assignment where t_{10} is 0. $c_1(b) = 0$ but $c_2(b) = 1$. $c_1(a) = 1$, but we don't know what $c_2(a)$ is. g_1 is an $\mathrm{ID}^{1,0}$-formula and g_2 is either a $\mathrm{NOT}^{1,0}$-formula or a $\mathrm{TRUE}^{1,0}$-formula. Thus, we get either a $(\mathrm{ID}, \mathrm{NOT})^{1,0}$-template or a $(\mathrm{ID}, \mathrm{TRUE})^{1,0}$-template, depending on whether $c_2(a)$ is false or true, respectively.
3. If both t_{01} and t_{10} appear, we have two possibilities. If c_1 can be satisfied with $t_{01} = 1$ (then t_{10} must be assigned 1 as well, since $c_1(b)$ is not true), we define the assignment a' to be equal to a except in indices of t_{01}, where a' will be 1. Using a' instead of a, we are now in case 2. If c_1 cannot be satisfied with $t_{01} = 1$, the only satisfying assignments to $c_1(t_{01}, t_{10})$ are ones where $t_{01} = 0$, while c_2 can be satisfied with $t_{01} = 1$.

The primary variable is t_{01} and t_{10} is an auxiliary variable. g_1 is a $\text{NOT}^{1,0}$-formula and g_2 is either a $\text{TRUE}^{1,0}$-formula or an $\text{ID}^{1,0}$-formula. This shows the existence of a $(\text{NOT}, \text{TRUE})^{1,0}$-template or a $(\text{NOT}, \text{ID})^{1,0}$-template, depending on whether c_2 can be satisfied with $t_{01} = 0$ or not, respectively.

◀

2.2 UFGs for All NP-Hard CSPs

In this section we prove Theorem 8.1. The proof follows from the next three lemmas.

▶ **Lemma 12.** *If there is an $(x \vee y \vee z, x \vee y \vee \bar{z}, \cdots, \bar{x} \vee \bar{y} \vee \bar{z})$-template over S, then S has a universal factor graph.*

Proof. Let H be a UFG for 3SAT (such as the $(1, 77/80)$-UFG for 3SAT from [6]). Replace the edges representing every 3CNF clause in H with the edges of the $(x \vee y \vee z, \cdots, \bar{x} \vee \bar{y} \vee \bar{z})$-template, where the auxiliary variables are new and unique to each constraint, and the primary variables are the same variables of the 3CNF clause. We get the factor graph H_G and it is universal; Given a formula φ that has factor graph H, we can instantiate in H_G each template corresponding to a 3CNF clause to represent the same 3CNF clause as in H. Hence, we get a formula that is equivalent to the satisfiability of φ.

Let C be the number of the constraint vertices in the $(x \vee y \vee z, x \vee y \vee \bar{z}, \cdots, \bar{x} \vee \bar{y} \vee \bar{z})$-template. Then, if φ is at most α-satisfiable, our instantiation of H_G is at most $\left(\alpha + \frac{(C-1)}{C}(1-\alpha)\right)$-satisfiable. ◀

▶ **Lemma 13** (see [9, 5])**.** *For every NP-hard constraint set S and every truth table f over a set of variables X, there is a $f(X)$-formula over S. Some of the (auxiliary) variables may be forced to be constants.*

▶ **Lemma 14.** *If a constraint set S is NP-hard, has two different satisfiable constraints, and we can use constants in place of variables, then there is an $(x \vee y \vee z, x \vee y \vee \bar{z}, \cdots, \bar{x} \vee \bar{y} \vee \bar{z})$-template over S.*

Proof. Suppose that $\tilde{t}$, the template given by Lemma 11, is $(\text{ID}, \text{TRUE})^{1,0}$.

Let f be the following truth table (over variables $x, y, z, s_{000}, s_{001}, s_{010}, s_{011}, s_{100}, s_{101}, s_{110}, s_{111}$):

$$(\overline{s_{000}} \vee x \vee y \vee z) \wedge (\overline{s_{001}} \vee x \vee y \vee \bar{z}) \wedge (\overline{s_{010}} \vee x \vee \bar{y} \vee z) \wedge (\overline{s_{011}} \vee x \vee \bar{y} \vee \bar{z}) \wedge$$

$$\wedge (\overline{s_{100}} \vee \bar{x} \vee y \vee z) \wedge (\overline{s_{101}} \vee \bar{x} \vee y \vee \bar{z}) \wedge (\overline{s_{110}} \vee \bar{x} \vee \bar{y} \vee z) \wedge (\overline{s_{111}} \vee \bar{x} \vee \bar{y} \vee \bar{z})$$

From Lemma 13, there is an f-formula over S. Add the template $\tilde{t}$ over each of the s variables (recall that the template $\tilde{t}$ has only one primary variable) with new and unique auxiliary variables for each copy of the template. We claim that the constructed graph is the $(x \vee y \vee z, x \vee y \vee \bar{z}, \cdots, \bar{x} \vee \bar{y} \vee \bar{z})$-template over S: Instantiate $\tilde{t}(s_u)$ to be $\text{ID}(s_u)$ and all other $\tilde{t}(s_v)$ to be $\text{TRUE}(s_v)$ for $v \neq u$. To satisfy the formula, s_u must be 1. s_v for $v \neq u$ can be 0, and setting $s_v = 0$ only improves the satisfiability of the formula. By setting the s variables in this way, the formula is satisfiable iff the clause on x, y, z corresponding to u is true (for example, if $u = 000$, then s_{000} must be 1 due to the constraint $\text{ID}(s_{000})$, setting all other s variables to 0 satisfies $\overline{s_v} \vee \cdots$ and $\text{TRUE}(s_v)$, so the formula is satisfied only when $x \vee y \vee z$ is true).

If $\tilde{t}$ is $(\text{NOT}, \text{ID})^{1,0}$ then the same proof works except we use $\text{NOT}(s_v)$ instead of $\text{TRUE}(s_v)$, which also means that the s_v's must be 0 (instead of can be 0).

If $\tilde{t}$ is $(\mathrm{NOT}, \mathrm{TRUE})^{1,0}$ we define f to be a similar truth table, except with the s variables non-negated. Using $\mathrm{NOT}(s_u)$ instead of $\mathrm{ID}(s_u)$ gives the same result, except s_u must be 0 and s_v should be 1. ◀

Proof of Theorem 8.1. Given an NP-hard constraint set S with at least two satisfiable constraints, if we are allowed to use the constants 0 and 1, using Lemmas 12 and 14 we have proven the Theorem. To simulate the usage of constants, we use the same method used in [9, 5]; we put a new variable z in all locations where we used the constant 0, and a new variable o in all locations we used the constant 1. From Lemma 13, there is a 2XOR-formula over S. We use o and z as the two primary variables of the 2XOR-formula, and give the constraints of the formula a large weight.

If there is a constraint c in S and an assignment a such that $c(a) = 1$ but $c(\bar{a}) = 0$, where $\bar{a}$ is the complement of a, then we add the constraint c and put o in all the indices for which a is one, and z in all other indices. We give c a large weight as well. If there is no such constraint, then for every formula over S, an assignment and its complement satisfy exactly the same constraints, thus we may assume WLOG that $o = 1$ and $z = 0$. ◀

2.3 UFGs for All APX-Hard CSPs

In this section we prove Theorem 8.2. The proof follows from the next four lemmas.

▶ **Lemma 15** (Lemma 5.37 in [5]). *For every* APX*-hard constraint set S there are constants $c^S > s^S$ and a $\mathrm{2XOR}^{c^S,s^S}$-formula over S.*

▶ **Lemma 16.** *There is a (κ, ς)-universal factor graph for maximum directed cut, for some $0 < \varsigma < \kappa < 1$.*

Proof. Let H_0 be a UFG for 3SAT (such as the $(1, 77/80)$-UFG for 3SAT from [6]). Using the standard gadget reduction (adding a single variable z to all clauses), we can transform H_0 to be H_1, a UFG for 4NAE. Using another standard gadget reduction (splitting each constraint $\mathrm{4NAE}(a, b, c, d)$ into two constraints $\mathrm{3NAE}(a, b, e)$ and $\mathrm{3NAE}(\bar{e}, c, d)$, where e is a new variable for each vertex), we transform H_1 into H_2, a UFG for 3NAE.

By transforming each constraint $\mathrm{3NAE}(a, b, c)$ into three linear equations $a \oplus b = 1$, $b \oplus c = 1$ and $c \oplus a = 1$ (here, since a, b, c are literals, the 1 may be changed into a 0, depending on the parity of the number of negations), we get three equations for each clause, where the clause is NAE-satisfied iff two of the equations are satisfied, and it is not NAE-satisfied iff none of the equations are satisfied. This transforms H_2 into H_3, a UFG for 2LIN.

Let $k\mathrm{LIN}^+$ be the set of constraints of kLIN with the addition of the constraint NOT (which does not have the arity k, but we may add inputs on which NOT does not depend). We add a single variable w to all equations, that is, every equation $x + y = b$ (for $b \in \{0, 1\}$) is transformed into an equation $x + y + w = b$ (and, conditioned on $w = 0$, every assignment satisfies the exact same set of constraints before and after the transformation). We also add the constraint $\mathrm{NOT}(w)$ (with large enough weight so satisfying it will always improve the number of satisfied constraints), and this transforms H_3 into H_4, a UFG for $3\mathrm{LIN}^+$.

Trevisan and al. [10] show (using their terminology) an optimal and strict 6.5-gadgets reducing PC_0 and PC_1 to DICUT. That is, they generate two directed weighted graphs on ten vertices, of which three are special and named x_1, x_2, x_3. In one graph, if $x_1 \oplus x_2 \oplus x_3 = 0$, the maximum cut has weight 13, otherwise the maximum cut has weight 12. In the other graph, if $x_1 \oplus x_2 \oplus x_3 = 1$, the maximum cut has weight 13, otherwise the maximum cut has weight 12. Moreover, the graphs are similar, except the directions of the edges are reversed

in one graph compared to the other. Using this gadget on every 3LIN formula, we transform H_4 into a UFG for MAXDICUT, except for the NOT (w) constraint. However, we can easily simulate this constraint, by adding a new vertex w' and an edge from w' to w with the same weight as the NOT (w) constraint (since w' has no incoming edges, the number of satisfied constraint cannot decrease by assigning it 1, so the edge is in the cut iff NOT (w)). ◀

The constraint $x \curvearrowright y$ means cutting an edge of a directed graph. That is, the constraint is satisfied iff $x = 1$ and $y = 0$.

▶ **Lemma 17.** *If there is an $(x \curvearrowright y, y \curvearrowright x)^{c,s}$-template over S, then S has a universal factor graph (without perfect completeness).*

Proof. Let H be a UFG for MAXDICUT from Lemma 16. Replace each edge of H with the edges of the $(x \curvearrowright y, y \curvearrowright x)^{c,s}$-template, where the auxiliary variables are new and unique to each constraint, and the two primary variables of the template are the variables of the original edge. We get the factor graph H_G and it it universal; Given a direction for the edges of the factor graph H, we can choose each of the $(x \curvearrowright y, y \curvearrowright x)^{c,s}$-templates to correspond to one of the directions of a directed edge. Let C be the number of the constraint vertices in the template. If the maximum directed cut has weight α, at most $\frac{\alpha c+(1-\alpha)s}{C}$ constraints of the corresponding formula over S can be satisfied, and exactly that amount can be satisfied, by the definition of a $(x \curvearrowright y, y \curvearrowright x)^{c,s}$-template. Therefore, if H is a (κ, ς)-UFG, H_G is a $\left(\frac{\kappa c+(1-\kappa)s}{C}, \frac{\varsigma c+(1-\varsigma)s}{C}\right)$-UFG for S. ◀

▶ **Lemma 18.** *If a constraint set S that contains two satisfiable constraints is* APX*-hard, and we can use constants in place of variables, then there is an $(x \curvearrowright y, y \curvearrowright x)^{c,s}$-template over S.*

Proof. Since S is APX-hard, there is a $2\text{XOR}^{c^S,s^S}$-formula over S. Let $\tilde{t}$ be the template given by Lemma 11 and suppose it is $(\text{ID}, \text{NOT})^{1,0}$. We can use the two formulas derived from this template and the $2\text{XOR}^{c^S,s^S}$-formula to implement $x \curvearrowright y$ by

$$2\text{XOR}\,(x, y) \wedge \text{ID}\,(x) \wedge \text{NOT}\,(y)$$

and we set formulas derived from $\tilde{t}$ (that is, ID and NOT) have weight $c^S - s^S$.

If $x \curvearrowright y$ is satisfied, then all constraints are satisfied, so the total weight of satisfied constraints is $3c^S - 2s^S$.

If $x = y$, then $2\text{XOR}\,(x, y)$ is unsatisfied and the $\text{ID}\,(x)$ or $\text{NOT}\,(y)$ constraint are satisfied, but not both, so the total weight of satisfied constraints is c^S. If $x = 0$ and $y = 1$, then $2\text{XOR}\,(x, y)$ is satisfied, but both the $\text{ID}\,(x)$ or $\text{NOT}\,(y)$ constraint are unsatisfied, so the total weight of satisfied constraints is c^S. Therefore, this implementation is a $x \to y^{c,s}$-formula, for $c = 3c^S - 2s^S$ and $s = c^S$.

$y \curvearrowright x$ can be implemented by

$$2\text{XOR}\,(x, y) \wedge \text{NOT}\,(x) \wedge \text{ID}\,(y)$$

and again, the constraints derived from $\tilde{t}$ have weight $c^S - s^S$. By the same argument, if $y \curvearrowright x$ is satisfied, then all constraints are satisfied, so the total weight of satisfied constraints is $3c^S - 2s^S$. If $y \curvearrowright x$ is unsatisfied, the total weight of satisfied constraints is c^S. Therefore, this implementation is a $y \curvearrowright x^{c,s}$-formula, for $c = 3c^S - 2s^S$ and $s = c^S$.

Since we have shown a $x \curvearrowright y^{c,s}$-formula and a $y \curvearrowright x^{c,s}$-formula with the same factor graph, we have a $(x \curvearrowright y, y \curvearrowright x)^{c,s}$-template over S in the case that $\tilde{t}$ is $(\text{ID}, \text{NOT})^{1,0}$.

If $\tilde{t}$ is $(\text{NOT}, \text{TRUE})^{1,0}$, we implement $x \curvearrowright y$ by

$$\text{2XOR}(x,y) \wedge \text{2XOR}(x,x') \wedge \text{2XOR}(y,y') \wedge \text{TRUE}(x) \wedge \text{NOT}(x') \wedge \text{NOT}(y) \wedge \text{TRUE}(y')$$

and $y \curvearrowright x$ by

$$\text{2XOR}(x,y) \wedge \text{2XOR}(x,x') \wedge \text{2XOR}(y,y') \wedge \text{NOT}(x) \wedge \text{TRUE}(x') \wedge \text{TRUE}(y) \wedge \text{NOT}(y')$$

where x' and y' are auxiliary variables. We set constraints derived from $\tilde{t}$ have weight $c^S - s^S$.

We now show that our implementation of $x \curvearrowright y$ satisfies the conditions of a $x \curvearrowright y^{c,s}$-formula, for specific c and s. If $x \curvearrowright y$ is satisfied, then all constraints are satisfied by setting $x' = 0, y' = 1$, so the total weight of satisfied constraints is $7c^S - 4s^S$.

If $x = y$, $\text{2XOR}(x,y)$ is not satisfied. We show that one more constraint must be unsatisfied, and we can ensure that only one more is unsatisfied. $\text{2XOR}(x,x')$ or $\text{NOT}(x')$ can be satisfied by changing x', without harming the other constraints, so there are three cases to consider:

1. $\text{2XOR}(x,x')$ and $\text{NOT}(x')$ are satisfied. Then, $0 = x' \neq x = y = 1$, so $\text{NOT}(y)$ is unsatisfied ($\text{2XOR}(y,y')$ is satisfied by setting $y' = 0$ without harming the other constraints). The total weight of satisfied constraints is $5c^S - 2s^S$.
2. $\text{2XOR}(x,x')$ is unsatisfied and $\text{NOT}(x')$ is satisfied. Then, $0 = x' = x = y$, so $\text{NOT}(y)$ is satisfied, and $\text{2XOR}(y,y')$ is satisfied by setting $y' = 0$. The total weight of satisfied constraints is $5c^S - 2s^S$.
3. $\text{2XOR}(x,x')$ is satisfied and $\text{NOT}(x')$ is unsatisfied. Then, $1 = x' \neq x = y = 0$, so $\text{NOT}(y)$ is satisfied, and $\text{2XOR}(y,y')$ is satisfied by setting $y' = 0$. The total weight of satisfied constraints is $5c^S - 2s^S$.

Lastly, if $x = 0, y = 1$, $\text{NOT}(y)$ is unsatisfied. By setting $y' = 0$ we satisfy $\text{2XOR}(y,y')$ without harming the other constraints. Since $x = 0$, either $\text{2XOR}(x,x')$ or $\text{NOT}(x')$ must be unsatisfied, so the total weight of satisfied constraints is $5c^S - 2s^S$.

Therefore, the implementation to $x \curvearrowright y$ is a $x \curvearrowright y^{c,s}$-formula, for $c = 7c^S - 4s^S$ and $s = 5c^S - 2s^S$. Similar arguments show that the implementation to $y \curvearrowright x$ is a $y \curvearrowright x^{c,s}$-formula, for the same c and s. Since we have shown a $x \curvearrowright y^{c,s}$-formula and a $y \curvearrowright x^{c,s}$-formula with the same factor graph, we have a $(x \curvearrowright y, y \curvearrowright x)^{c,s}$-template over S in the case that $\tilde{t}$ is $(\text{NOT}, \text{TRUE})^{1,0}$.

Finally, in the case that $\tilde{t}$ is $(\text{ID}, \text{TRUE})^{1,0}$, we implement $x \curvearrowright y$ by

$$\text{2XOR}(x,y) \wedge \text{2XOR}(x,x') \wedge \text{2XOR}(y,y') \wedge \text{ID}(x) \wedge \text{TRUE}(x') \wedge \text{TRUE}(y) \wedge \text{ID}(y')$$

and $y \curvearrowright x$ by

$$\text{2XOR}(x,y) \wedge \text{2XOR}(x,x') \wedge \text{2XOR}(y,y') \wedge \text{TRUE}(x) \wedge \text{ID}(x') \wedge \text{ID}(y) \wedge \text{TRUE}(y')$$

and we set constraints derived from $\tilde{t}$ have weight $c^S - s^S$. Similar arguments as before show that we have a $x \curvearrowright y^{c,s}$-formula and a $y \curvearrowright x^{c,s}$-formula, for $c = 7c^S - 4s^S$ and $s = 5c^S - 2s^S$. Hence, there is a $(x \curvearrowright y, y \curvearrowright x)^{c,s}$-template over S in this case as well. ◀

Proof of Theorem 8.2. Given an APX-hard constraint set S, with at least two satisfiable constraints, if we are allowed to use the constants 0 and 1, Lemmas 17 and 18 show that S has a (κ, ς)-UFG (for some constants $1 \geq \kappa > \varsigma \geq 0$). In order to bypass the use of the constants 0 and 1, we use the same method as in the proof of Theorem 8.1 to simulate the constants (and we use Lemma 15 to show the existence of a 2XOR-formula).

What remains to prove the theorem is to handle the case that S has two constraints but one of them is FALSE. Since 2XOR can always be implemented by an APX-hard constraint set, and we have a constraint that is never satisfied, we have a $(\text{2XOR}, \text{FALSE})^{c,s}$-template (in the second case, we only use the FALSE constraint). By replacing all edges of a clique on n vertices by this constraint we get the required UFG; every 2XOR instance over n variables can be represented by a subset E' of the edges of the clique on n vertices. By setting the all $(\text{2XOR}, \text{FALSE})^{c,s}$-template that correspond to edges in E' to be 2XOR, and all others to be FALSE, we get a formula over S corresponding to the 2XOR instance. ◀

Acknowledgments. Work supported in part by the Israel Science Foundation (grant No. 621/12).

References

1. Per Austrin and Johan Håstad. On the usefulness of predicates. *TOCT*, 5(1):1, 2013.
2. Jehoshua Bruck and Moni Naor. The hardness of decoding linear codes with preprocessing. *Information Theory, IEEE Transactions on*, 36(2):381 –385, March 1990.
3. Andrei A. Bulatov. On the CSP dichotomy conjecture. In *Computer Science – Theory and Applications*, volume 6651 of *Lecture Notes in Computer Science*, pages 331–344. Springer, 2011.
4. Nadia Creignou. A dichotomy theorem for maximum generalized satisfiability problems. *J. Comput. Syst. Sci.*, 51(3):511–522, 1995.
5. Nadia Creignou, Sanjeev Khanna, and Madhu Sudan. *Complexity classifications of boolean constraint satisfaction problems*. Society for Industrial and Applied Mathematics, 2001.
6. Uriel Feige and Shlomo Jozeph. Universal factor graphs. In *ICALP*, pages 339–350, 2012.
7. Subhash Khot, Preyas Popat, and Nisheeth K. Vishnoi. $2^{\log^{1-\epsilon} n}$ hardness for the closest vector problem with preprocessing. In *STOC'12*, pages 277–288, 2012.
8. Subhash Khot, Madhur Tulsiani, and Pratik Worah. A characterization of strong approximation resistance. In *STOC'14*, 2014.
9. Thomas J. Schaefer. The complexity of satisfiability problems. In *STOC'78*, pages 216–226. ACM, 1978.
10. Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson. Gadgets, approximation, and linear programming. *SIAM Journal on Computing*, 29(6):2074–2097, 2000.

A $\frac{9}{7}$-Approximation Algorithm for Graphic TSP in Cubic Bipartite Graphs *

Jeremy A. Karp[1] and R. Ravi[2]

1 **Tepper School of Business, Carnegie Mellon University**
5000 Forbes Ave, Pittsburgh, PA, USA
jkarp@andrew.cmu.edu

2 **Tepper School of Business, Carnegie Mellon University**
5000 Forbes Ave, Pittsburgh, PA, USA
ravi@andrew.cmu.edu

Abstract

We prove new results for approximating Graphic TSP. Specifically, we provide a polynomial-time $\frac{9}{7}$-approximation algorithm for cubic bipartite graphs and a $(\frac{9}{7} + \frac{1}{21(k-2)})$-approximation algorithm for k-regular bipartite graphs, both of which are improved approximation factors compared to previous results. Our approach involves finding a cycle cover with relatively few cycles, which we are able to do by leveraging the fact that all cycles in bipartite graphs are of even length along with our knowledge of the structure of cubic graphs.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Approximation algorithms, traveling salesman problem, Barnette's conjecture, combinatorial optimization

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.284

1 Introduction

1.1 Motivation and Related Work

The traveling salesman problem (TSP) is one of most well known problems in combinatorial optimization, famous for being hard to solve precisely. In this problem, given a complete undirected graph $G = (V, E)$ with vertex set V and edge set E, with non-negative edge costs $c \in \mathbb{R}^{|E|}$, $c \neq 0$, the objective is to find a Hamiltonian cycle in G of minimum cost. In its most general form, TSP cannot be approximated in polynomial time unless $P = NP$. In order to successfully find approximate solutions for TSP, it is common to require that instances of the problem have costs that satisfy the triangle inequality ($c_{ij} + c_{jk} \geq c_{ik}\ \forall i, j, k \in V$). This is the Metric TSP problem. The Graphic TSP problem is a special case of the Metric TSP, where instances are restricted to those where $\forall i, j \in E$, the cost of edge (i, j) in the complete graph G are the lengths of the shortest paths between nodes i and j in an unweighted, undirected graph, on the same vertex set.

One value related to the ability to approximate TSP is the integrality gap, which is the worst-case ratio between the optimal solution for a TSP instance and the solution to a linear programming relaxation called the subtour relaxation [7]. A long-standing conjecture (see, e.g.,[11]) for Metric TSP is that the integrality gap is $\frac{4}{3}$. One source of motivation for studying Graphic TSP is that the family of graphs with two vertices connected by three paths

* Supported in part by NSF grant CCF-1218382

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 284–296

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

of length k has an integrality gap that approaches $\frac{4}{3}$. This family of graphs demonstrates that Graphic TSP captures much of the complexity of the more general Metric TSP problem.

For several decades, Graphic TSP did not have any approximation algorithms that achieved a better approximation than Christofides' classic $\frac{3}{2}$-approximation algorithm for Metric TSP [4], further motivating the study of this problem. However, a wave of recent papers [9, 1, 3, 10, 13, 5, 14] have provided significant improvements in approximating Graphic TSP. Currently, the best known approximation algorithm for Graphic TSP is due to Sebő and Vygen [14], with an approximation factor of $\frac{7}{5}$.

Algorithms with even smaller approximation factors have also been found for Graphic TSP instances generated by specific subclasses of graphs. In particular, algorithms for Graphic TSP in cubic graphs (where all nodes have degree 3) have drawn significant interest as this appears to be the simplest class of graphs that has many of the same challenges as the general case. Currently, the best approximation algorithm for Graphic TSP in cubic graphs is due to Correa, Larré, and Soto [5], whose algorithm achieves an approximation factor of $(\frac{4}{3} - \frac{1}{61236})$ for 2-edge-connected cubic graphs. Progress in approximating Graphic TSP in cubic graphs also relates to traditional graph theory, as Barnette's conjecture [2] states that all bipartite, planar, 3-connected, cubic graphs are Hamiltonian. This conjecture suggests that instances of Graph TSP on Barnette graphs could be easier to approximate, and conversely, approximation algorithms for Graphic TSP in Barnette graphs may lead to the resolution of this conjecture. Indeed, Correa, Larré, and Soto [6] provided a $(\frac{4}{3} - \frac{1}{18})$-approximation algorithm for Barnette graphs. Along these lines, Aggarwal, Garg, and Gupta [1] were able to obtain a $\frac{4}{3}$-approximation algorithm for 3-edge-connected cubic graphs before any $\frac{4}{3}$-approximation algorithms were known for all cubic graphs. In this paper, we examined graphs that are cubic and bipartite, another class of graphs that includes all Barnette graphs. An improved approximation for this class of graphs is the primary theoretical contribution of this paper:

▶ **Theorem 1.** *Given a cubic bipartite connected graph G with n vertices, there is a polynomial time algorithm that computes a spanning Eulerian multigraph H in G with at most $\frac{9}{7}n$ edges.*

▶ **Corollary 2.** *Given a k-regular bipartite connected graph G with n vertices where $k \geq 4$, there is a polynomial time algorithm that computes a spanning Eulerian multigraph H in G with at most $(\frac{9}{7} + \frac{1}{21(k-2)})n - 2$ edges.*

Proof. First, we will extract k edge-disjoint perfect matchings from G. We find a cubic subgraph, G_{cubic}, by taking the union of any two of these perfect matchings and the perfect matching (of the remaining $k-2$ matchings) such that the number of $K_{3,3}$ components in the resulting subgraph is minimal. For a detailed description of this procedure, see [12, Algorithm 9]. In each component of G_{cubic} that is a $K_{3,3}$ we will find a 6-cycle covering these nodes. This can be done in constant time by taking any walk through this component that does not visit a node twice as long as this is possible, then returning to the first node. In every other connected component, run the Compress and Expand phases of the BIGCYCLE algorithm, which will find a 2-factor over each component containing at most $\frac{n_i}{7}$ cycles, where n_i is the number of nodes in the connected component. We apply the pigeonhole principle – the $K_{3,3}$ components contained in any of the $k-2$ cubic subgraphs from which we selected G_{cubic} are the "pigeons", of which there are at most at most $\frac{n}{6}$, and these $k-2$ cubic subgraphs are the "holes" – and conclude that there are x_1 nodes in $K_{3,3}$s within G_{cubic}, where $x_1 \leq \frac{n}{k-2}$ [12, Lemma 7]. These nodes are covered by $\frac{x_1}{6}$ cycles. Then, there x_2 nodes in the remaining components and $x_1 + x_2 = n$. These x_2 nodes are covered by at most $\frac{x_2}{7}$ cycles. Then, the overall 2-factor of G has at most $\frac{x_1}{6} + \frac{x_2}{7}$ cycles. The following

calculations compute an upper bound on these cycles in terms of n:

$$\begin{aligned}\frac{x_1}{6} + \frac{x_2}{7} &= \frac{7x_1 + 6x_2}{42} \\ &= \frac{6n + x_1}{42} \\ &= \frac{n}{7} + \frac{x_1}{42} \\ &\leq \frac{n}{7} + \frac{n}{42(k-2)}\end{aligned}$$

By Proposition 3, this 2-factor can be extended into a spanning Eulerian multigraph in G with at most $n + 2(\frac{n}{7} + \frac{n}{42(k-2)} - 1) = (\frac{9}{7} + \frac{1}{21(k-2)})n - 2$ edges, proving the corollary. ◀

This result complements results [15, 8] which provide guarantees for k-regular graphs in the asymptotic regime. Corollary 2 improves on these guarantees for small values of k.

1.2 Overview

In this paper, we will present an algorithm to solve Graphic TSP, which guarantees a solution with at most $\frac{9}{7}n$ edges in cubic bipartite graphs. The best possible solution to Graphic TSP is a Hamiltonian cycle, which has exactly n edges, so this algorithm has an approximation factor of $\frac{9}{7}$.

A corollary of Petersen's theorem is that every cubic bipartite graph contains three edge-disjoint perfect matchings. The union of any 2 of these matchings forms a 2-factor. The following proposition demonstrates the close relationship between 2-factors and Graphic TSP tours in connected graphs.

▶ **Proposition 3.** *Any 2-factor with k cycles in a connected graph can be extended into a spanning Eulerian multigraph with the addition of exactly $2(k-1)$ edges. This multigraph contains exactly $n + 2(k-1)$ edges in total.*

Proposition 3 can be implemented algorithmically by compressing each cycle into a single node and then finding a spanning tree in this compressed graph. We then add two copies of the edges from this spanning tree to the 2-factor. We present an algorithm, BIGCYCLE, which begins by finding a 2-factor with at most $\frac{n}{7}$ cycles. Then, it applies Proposition 3 to generate a spanning Eulerian subgraph from this 2-factor containing at most $n + 2 \times (\frac{n}{7} - 1) = \frac{9}{7}n - 2$ edges.

BIGCYCLE first shrinks every 4-cycle in the graph, then it generates a 2-factor in the condensed graph. If the resulting 2-factor has no 6-cycles, then we can expand the 4-cycles and this will be our solution. If the 2-factor does have a 6-cycle, then the algorithm contracts either this 6-cycle or a larger subgraph that includes this 6-cycle. We are able to iterate this process until we find a 2-factor in the compressed graph with no "organic" 6-cycles. At this point, the algorithm is able to expand the compressed graph back to its original state, maintaining a 2-factor with relatively few cycles. Theorem 13 in Section 3.5 proves that this 2-factor has at most $\frac{n}{7}$ cycles.

2 A $\frac{9}{7}$-Approximation Algorithm for Graphic TSP in Cubic Bipartite Graphs

2.1 Overview

In a graph with no 4-cycles (squares), all 2-factors will have an average cycle length of at least 6, so all 2-factors will have at most $\frac{n}{6}$ cycles, which results in a $\frac{4}{3}$-approximation. In order to improve our approximation guarantee, we need to target 6-cycles, as well as 4-cycles. The algorithm we present finds a square-free 2-factor in which every 6-cycle can be put in correspondence with a distinct cycle of size 8 or larger. Then, we can find a 2-factor in which every large cycle and its corresponding 6-cycles have average cycle length of at least 7 via an amortized analysis over the compressing iterations (Lemma 12 in Section 3.4). We then show that this is enough to conclude that the 2-factor contains at most $\frac{n}{7}$ cycles (Theorem 13 in Section 3.5). This is primary contribution of this paper.

A method used throughout this paper is to systematically replace certain subgraphs containing 4-cycles and 6-cycles with other subgraphs. We will refer to these replacement subgraphs as "gadgets". To keep track of portions of the graph that have not been altered by these gadgets, we define the term "organic" as follows:

▶ **Definition 4.** A subgraph is organic if it consists entirely of nodes and edges contained in the original graph. For a single edge to be organic, both its end-nodes must be organic.

We also give a formal definition of the term "gadget":

▶ **Definition 5.** A gadget is a subgraph that is inserted into the graph by the BIGCYCLE algorithm in place of a different subgraph. Examples of gadgets are shown in Figures 2, 4, 6, 8, and 10. Gadgets are used to replace other subgraphs containing 4- or 6-cycles.

In the following section, we introduce the gadgets used in the BIGCYCLE algorithm. When our 2-factor contains 4-cycles and organic 6-cycles, we will use these gadgets to condense our graph and remove these cycles. We then repeat this process (condensing 4-cycles that appear along the way) and compute a new 2-factor in the condensed graph until we obtain a 2-factor with no organic 6-cycles. We show that expanding the graph can create a small number of new 6-cycles in our 2-factor, but we are able to account for them, ensuring that the bounds described in the previous paragraph must hold.

2.2 Gadgets

In this section, we present the most important subgraphs that will be replaced with gadgets by the algorithm. Several of the more specialized gadgets are omitted in this article. All gadgets are shown in the extended version of this paper [12]. In total, there are 3 gadgets to replace 4-cycles and 6 gadgets to replace 6-cycles. We will give these configurations the names S_1, S_2, S_3, H_1, H_2, H_3, H_4, H_5, and H_6. The gadget that replaces a configuration H_i will be called H_i'.

First, we introduce the gadget we use to replace squares whose outgoing edges are incident on four distinct vertices

We also introduce the S_3, which is used to replace squares whose outgoing edges are incident on only two vertices.

The S_2, another square-replacing gadget is omitted. The first gadget used to replace 6-cycles is two super-vertices which replace a simple 6-cycle, H_1.

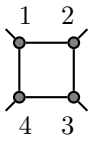


Figure 1 A square with four distinct neighbors: S_1.

Figure 2 The gadget that replaces this configuration: S_1'.

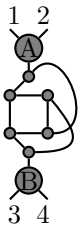


Figure 3 A square with two distinct neighbors: S_3.

Figure 4 The super-edge which replaces this configuration: S_3'.

The remaining gadgets are special cases of 6-cycles. In this paper, only the first two of these specialized gadgets are displayed. Note that every H_2 contains a H_1, all H_3s contain a H_2, and H_4s, H_5s, and H_6s are special cases of H_3s.

The motivation to use these additional gadgets comes out of necessity, to prevent large numbers of 6-cycles from being introduced into the 2-factor during the expansion phase of the algorithm. For example, Figures 13 and 14 in Section 3 document an expansion that turns an $x + y + 4$-cycle in the cycle cover passing through a gadget which replaced a H_1 into two cycles of lengths $x + 3$ and $y + 5$. Since the algorithm will condense H_2s before H_1s, this ensures that y, the length of a path, is at least 3, meaning that the $y + 5$-cycle is not a 6-cycle. The motivation for introducing the remaining specialized gadgets is similar.

In the next subsection, we present a detailed description of the algorithm.

2.3 The Algorithm

Listing 1 presents pseudocode for the BIGCYCLE algorithm. The remainder of this section explains the details of the algorithm, broken up into three subroutines, and presents motivation for the operations performed by the algorithm. The COMPRESS, EXPAND, and DOUBLETREE subroutines called by BIGCYCLE are described in the following three subsections.

2.3.1 Finding a "Good" 2-factor in the Condensed Graph G_k

We start the algorithm by receiving a connected cubic bipartite graph. Call this graph G_0. If G_0 is a $K_{3,3}$ then we compute a 2-factor in this graph, which will be a Hamiltonian cycle,

Listing 1 BIGCYCLE(G)

```
Input: An undirected, unweighted, cubic, bipartite graph G = (V, E)
F_compressed ←COMPRESS(G)
F ←EXPAND(F_compressed)
TSP ←DOUBLETREE(G, F)
Return TSP
```

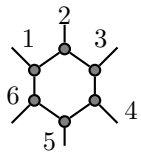


Figure 5 A simple 6-cycle: H_1.

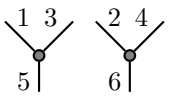


Figure 6 The gadget which replaces the 6-cycle: H_1'.

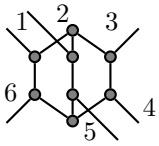


Figure 7 Two 6-cycles with 3 common edges: H_2.

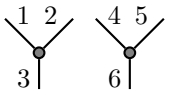


Figure 8 The gadget which replaces the configuration in Figure 7: H_2'.

and return this cycle as our solution. Otherwise, we search for 4-cycles that are not contained in $K_{3,3}$s and replace them with their corresponding gadgets until we are returned a graph with no squares except possibly inside of $K_{3,3}$s. Let i be the number of square compressions made, and let G_i be the compressed graph at the end of this process. Next, construct a 2-factor, F_i, in G_i. When we construct 2-factors throughout the algorithm, we do so by decomposing the graph into 3 edge-disjoint perfect matchings and taking the union of the two perfect matching containing the fewest S_3' gadgets, shown in Figure 4. These two perfect matchings form a 2-factor with limited potential to introduce organic 6-cycles of the type shown in Figure 16 (Section 3.2). If F_i contains no organic 6-cycles, then we advance to the next phase of the algorithm, described in the next subsection. In this case, $k = i$.

If F_i does contain an organic 6-cycle, C, then we check if the current compressed graph G_i contains organic subgraphs that can be replaced by gadgets in the following order (ordered from most specialized to most general): H_6, H_5, H_4, H_3, H_2, H_1. We choose the first organic configuration on the list (the most specialized configuration) we can find in G_i and replace this configuration with the corresponding gadget, outputting graph G_{i+1} to reflect this change. The order of choosing subgraphs to replace is useful in accounting for the average length of the cycles in the final 2-factor, as shown in the proof of Lemma 11. We then search for 4-cycles that are not contained in $K_{3,3}$s and replace them with their corresponding gadgets until we have removed any 4-cycles generated as a consequence of replacing a subgraph with one of our gadgets, obtaining a new compressed graph G_j, where $j - i + 1$ is the number of 4-cycles compressed. We construct a new 2-factor F_j and repeat the process in this paragraph until we have a 2-factor F_k with no organic 6-cycles, in a condensed graph G_k, where k is the total number of gadget replacement operations performed during this phase of the algorithm. This process is performed by the COMPRESS subroutine in Listing 1.

2.3.2 Expanding a 2-factor in G_k into a 2-factor in G_0

We will describe the process of expanding F_k and G_k so that we get back to the original graph G_0 with a desirable 2-factor F_0 in more detail.

We will reverse the process described in the previous subsection by replacing our gadgets in compressed graph G_i with the original configuration from the earlier graph G_{i-1} in the reverse order of that in which we replaced the configurations. In other words, the gadgets we

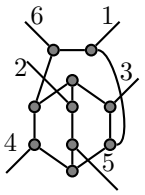


Figure 9 A specialized configuration containing three overlapping 6-cycles: H_3.

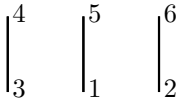


Figure 10 The gadget which replaces the configuration in Figure 9: H_3'.

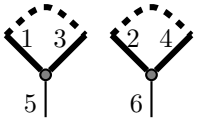


Figure 11 A pair of super-vertices in G_i. The bold edges are included in 2-factor F_i. The dashed bold edges represent a path, included in F_i.

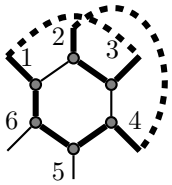


Figure 12 2-factor F_{i-1} after expanding the super-vertices from Fig. 11.

inserted last are those which we first replace with their original configuration. We call this process "expanding" because each one of these operations adds vertices and edges to the graph. After we have made each replacement to expand the graph, F_i is no longer a 2-factor in G_{i-1} because the new nodes added by the most recent expansion step are not covered by F_i. However, we can add edges to F_i so that it becomes a 2-factor, F_{i-1} in the graph after this expansion step. It may not be immediately clear that this is always possible. In fact, one of the bigger challenges in developing this algorithm was choosing a set of gadgets where this property holds. Figures 11 and 12 show an example of how this process works.

At each expansion, we are able to extend F_i into a set of edges F_{i-1}, which will be a 2-factor in the expanded graph, G_{i-1}. In order to optimize the performance of BIGCYCLE, we must impose one extra operation in this phase of the algorithm. After each expansion of a H_1' that introduces an organic 6-cycle, C_1, into the 2-factor, we will perform a local search to see if C_1 and the nearby edges of the newly expanded 2-factor F_{i-1}, those contained in neighborhood of nodes within distance 3 of C_1, can be altered in a way that reduces the number of cycles in the 2-factor and does not add any 4- or 6-cycles to the 2-factor . If this is possible, then we update F_{i-1} so that it contains fewer cycles. In addition to being an effective heuristic to reduce the number of cycles in our final 2-factor, this operation allows us to improve our approximation factor by eliminating an otherwise troubling corner case.

At this point, we can repeat the process of replacing gadgets with their original configurations and adding edges to the 2-factor until we have expanded the graph back to the original input G_0 and have a 2-factor, F_0, in this graph. This process is performed by the EXPAND subroutine in Listing 1.

2.3.3 Obtaining a Good Final Solution by Adding Edges to F_0

We now have a 2-factor F_0, which contains at most k cycles. We compress each cycle into a single node and compute a spanning tree in this compressed graph. This spanning tree has $k-1$ edges. Then, we add two copies of the edges in this spanning tree to our 2-factor F_0

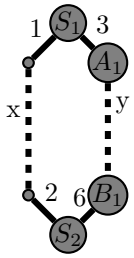


Figure 13 A cycle in F_i, a 2-factor over the condensed graph G_i. S_1 and S_2 are two super-vertices which replaced a standard hexagon. The dashed lines represent paths of length 3.

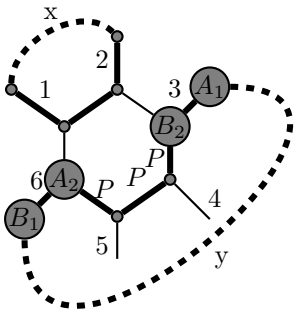


Figure 14 The cycle from Figure 13, after expanding S_1 and S_2

to obtain a solution with $n + 2(k-1)$ edges. In Section 3 we prove that F_0 has at most $\frac{n}{7}$ cycles, so this gives us a solution of at most $\frac{9}{7}n - 2$ edges. This process is performed by the DOUBLETREE subroutine in Listing 1.

3 Accounting for 6-Cycles

In the proof of our approximation guarantee, the limitation on producing a lower approximation factor comes from the possibility that some proportion of our final 2-factor's cycles will be of length 6. Most operations the algorithm performs while expanding the 2-factor from the condensed to the original graph result in cycles of length 8 or larger, so in this section we will look at operations that create organic 6-cycles in detail. To account for 6-cycles, we show that every organic 6-cycle can be put in correspondence with some long cycle of length 8 or longer. Then, Lemma 12 demonstrates that the average cycle length of any long cycle and its corresponding set of 6-cycles is sufficiently long to ensure that our final cycle cover has relatively few cycles, even if some of them are 6-cycles.

Figures 13 and 14, taking the dashed lines to be paths of lengths x and y, demonstrate how a $(y+7)$-cycle can turn into a 6-cycle and a $(y+5)$-cycle after an expansion if $x = 3$.

We will carefully analyze this and several other cases that form the bottleneck in our analysis, which occur when expanding our graph back to its original state. We will account for organic 6-cycles by creating a correspondence from every 6-cycle in the final 2-factor F_0 to some larger cycle in F_0. This way, if we can show that every large cycle of length $l \geq 8$ in F_0 is affiliated with at most $f(l)$ 6-cycles, then the average cycle length is at least $\min_{l \geq 8} \frac{6 \times f(l) + l}{f(l)+1}$. Once we have placed a lower-bound on the average cycle length in this manner (Lemma 12), we can easily determine our approximation factor (Theorems 1 and 13).

3.1 An Expansion that Introduces Organic 6-cycles

In this section we discuss the expansion shown in Figures 13 and 14, a representative example of how it is possible for a small number of 6-cycles to be included in the final 2-factor. This expansion is the only type of operation involving the H_1' gadget that can introduce an organic 6-cycle into the 2-factor during an expansion. There are two other expansions with a similar outcome which involve the H_2' and H_3' gadgets, respectively. The analysis to account for these expansions is very similar to the analysis of this case. Furthermore, these "bad" H_2 and H_3

expansions introduce larger sets of protected edges than the H_1 expansion in Figures 13 and 14, so this H_1 expansion is what limits the approximation factor we obtain in our analysis.

▶ Remark. If an expansion operation of the type shown in Figures 13 and 14 were to occur, then it is not possible for the nodes corresponding to A_1 and B_1 in these figures to be the super-vertices of a H_1's gadget whose expansion introduces an organic 6-cycle into the 2-factor. A proof of this remark can be found in the extended version of this paper [12, Remark 1].

We now give a definition and a lemma about "protected edges", organic paths which help us analyze the performance of the BIGCYCLE algorithm. We use this term because protected edges cannot be separated from each other in the 2-factor during subsequent expansion operations.

▶ **Definition 6.** Protected edges are edges contained in maximal paths that are organic, included in a cycle of length at least 8 in a 2-factor F_i, and part of an organic subgraph that was previously contracted and then expanded. Protected edges are identified during the "expanding" phase of the algorithm (described in Section 2.3), when the expansion operation introduces an organic 6-cycle. The edges labeled "P" in Figure 14 are an example of a set of protected edges.

▶ **Lemma 7.** *Let P_1 and P_2 be the sets of protected edges corresponding to two 6-cycles, C_1 and C_2, respectively. Then $P_1 \cap P_2 = \emptyset$ and $V(P_1) \cap V(P_2) = \emptyset$.*

▶ **Definition 8.** For a given 6-cycle, C, in the final 2-factor, F_0, consider the value i such that C is a cycle of F_i but not of F_{i+1}. Then, it was the expansion operation from G_{i+1} to G_i which "finalized" this cycle. Then, the protected edges identified during this "finalizing" operation are defined to be the protected edges corresponding to C. For example, in Figure 14 we put the edges labeled "P' in correspondence with the 6-cycle containing edges 1 and 2.

▶ **Definition 9.** For any cycle, C_i, of length at least 8, in the 2-factor F we will define a set of 6-cycles, S_{C_i} which correspond to C_i. For each 6-cycle, C, in F, we say C is an element of S_{C_i} if C's protected edges are in C_i or if C's protected edges are in another 6-cycle C' whose protected edges are in C_i.

3.2 Expanding Gadgets that Replaced Squares

Expanding S_3's, not in the 2-factor, of the type shown in Figure 15 can introduce organic 6-cycles into the 2-factor. However, we limit the number of S_3's whose edges are not in the 2-factor by computing three disjoint perfect matchings and taking as our 2-factor the union of the two perfect matchings that contain the most edges in S_3's. Then, at most $\frac{1}{3}$ of the S_3's will have edges not included in the 2-factor. Then, we can put the 6-cycles introduced by expanding each S_3' of this type in correspondence with a larger cycle containing the protected edges of the same type shown in Figure 16. These edges are labeled "P" in Figure 16.

3.3 Expanding All Other Gadgets

In the analysis that follows, we have identified all expansions with the potential to introduce organic 6-cycles cycles into F_i that were not present in F_{i+1}. Confirming this fact requires checking all possible ways a 2-factor can pass through each gadget to ensure that all expansions that can introduce organic 6-cycles are properly analyzed. Diagrams documenting every one of these other expansions are in the extended version of this paper [12], and a careful

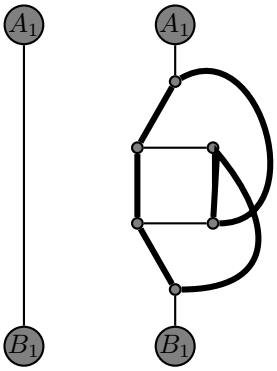


Figure 15 A S_3' in G_i which is not covered by the 2-factor F_i (left), and the S_3 in G_{i-1} that replaced the S_3' after expansion (right)

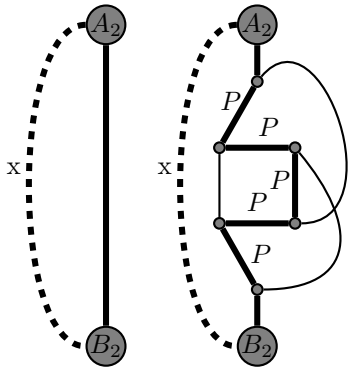


Figure 16 A S_3' in G_i that is covered by the 2-factor F_i (left), and the S_3 in G_{i-1} that replaced the S_3' after expansion (right). The 5 bold edges labeled "P" will be the 6-cycle in Figure 15's protected edges.

examination of these sections confirms that all unmentioned expansion operations are not capable of introducing an organic 6-cycle into the 2-factor. These other operations result either in converting one cycle into one larger cycle, converting one cycle into two large (length of at least 8) cycles, or converting two cycles into one or two large cycles.

3.4 Analyzing the Worst Case

We call the edges identified in Definition 6 "protected" because they form organic paths in our 2-factor, so these paths will remain part of the 2-factor regardless of the future expansion operations performed. In Definition 8, we established a correspondence between protected edges and 6-cycles in our 2-factor. This relation allows us to place every 6-cycle in correspondence with some large cycle in our 2-factor, as stated in Definition 9. We will use this correspondence to analyze the performance of BIGCYCLE. The proof of the following lemma is in the extended version of this paper [12, Lemmas 3 and 4].

▶ **Lemma 10.** *If a* 6*-cycle,* C_1*, has its corresponding protected edges in another* 6*-cycle,* C_2*, then* C_2 *has* 5 *corresponding protected edges. These protected edges are all in a cycle which either contains no other protected edges or is of length at least* 10*. Furthermore,* C_1 *and* C_2 *must have been introduced into the* 2*-factor during the expansion of a* H_1' *and* H_2'*, respectively.*

Before we prove a lower bound on average cycle length, we need the following lemma regarding 8-cycles in the final 2-factor.

▶ **Lemma 11.** *Every cycle* C *of length* 8 *in the final* 2*-factor* F *has at most one corresponding* 6*-cycle.*

This lemma is proved by examining the different ways in which two or more 6-cycles could be placed into correspondence with C and demonstrating that all of these cases result in a contradiction. The contradictions primarily arise from requiring C to contain more protected edges than it can fit. We then show that the remaining cases require the algorithm to have performed in a way that contradicts its instructions, such as compressing a H_2 when a square was in the graph, or combining two organic paths in separate cycles into a single cycle

separated by at most a single edge, something the BIGCYCLE algorithm will never do. The proof of this lemma is available in the extended version of this paper [12, Lemma 5].

We are now prepared to prove the next lemma, regarding average cycle length of a large cycle and its set of corresponding 6-cycles:

▶ **Lemma 12.** *For any cycle C in the final 2-factor F of length l such that $l \geq 8$ and its set of corresponding 6-cycles, the average length of this set of cycles is at least 7.*

Proof. First, consider the simple case where C has no corresponding 6-cycles. The set of cycles we are considering in this case is just a single cycle of length $l \geq 8$. $l \geq 8 > 7$, so in this case, the only cycle in the set of cycles has length at least 7.

Now, consider the case when all of the corresponding 6-cycles have their protected edges contained in the large cycle C (to be clear, the only way this condition could be violated is if some 6-cycle has its protected edges in another 6-cycle, whose protected edges are contained in C). Each of the expansion operations that included one of the corresponding 6-cycles in the 2-factor protects at least 3 edges, and these protected edges are contained in C, so at most $\frac{l}{3}$ 6-cycles can correspond to cycle C. If $l \geq 10$, then the average cycle length among cycle C and its corresponding 6-cycles is at least

$$\frac{\lfloor \frac{l}{3} \rfloor \times 6 + l}{\lfloor \frac{l}{3} \rfloor + 1} \geq 7$$

If $l = 8$ then by Lemma 11, C has at most one corresponding 6-cycle, so the average length of C and its corresponding 6-cycle is also 7. We must consider the case when at least one corresponding 6-cycle, C_1, has its protected edges in another 6-cycle, C_2. If $l = 8$ and C contains C_2's protected edges then C has at least two corresponding 6-cycles, C_1 and C_2, contradicting Lemma 11.

Next, consider if $l = 10$ and C contains C_2's 5 protected edges in the final 2-factor. C cannot contain another set of 5 protected edges, due to Lemma 7, because this would require these protected edges to share a node with C_2's protected edges. Then, in addition to C_1 and C_2, C can have at most one additional corresponding 6-cycle, otherwise C would contain more than 10 protected edges. In this case, C has at most 3 corresponding 6-cycles, so the average length of C and its corresponding 6-cycles is at most $\frac{10+3\times6}{4} = 7$.

The only remaining case is when $l \geq 12$ and at least one corresponding 6-cycle, C_1, has its protected edges contained in another 6-cycle, C_2. By Lemma 10, each 6-cycle has at least 3 protected edges in C or its protected edges are in another 6-cycle whose 5 protected edges are in C. So, if a corresponding 6-cycle's protected edges are not in C, then there is another 6-cycle corresponding to C for which these two 6-cycles contribute 5 protected edges to C. Then, each 6-cycle on average contributes at least $\frac{5}{2}$ protected edges to C, so there are at most $\frac{2l}{5}$ 6-cycles corresponding to C. Then, the average cycle length among cycle C and its corresponding 6-cycles is at least

$$\frac{\lfloor \frac{2l}{5} \rfloor \times 6 + l}{\lfloor \frac{2l}{5} \rfloor + 1} \geq 7 \text{ (because } l \geq 12)$$

In all possible cases, C and its corresponding 6-cycles have average length of at least 7. ◀

3.5 Main Theorems

▶ **Theorem 13.** *Given a cubic bipartite graph G with $n > 6$ vertices, there is a polynomial time algorithm that computes a 2-factor with at most $\frac{n}{7}$ cycles*

Proof. It follows directly from Lemma 12 that the average cycle length in the 2-factor produced by BIGCYCLE is at least 7. Then, it must be the case that BIGCYCLE produces a 2-factor with at most $\frac{n}{7}$ cycles.

The BIGCYCLE algorithm performs $O(n)$ contractions and expansions. In between each contraction, the algorithm will search for other subgraphs to contract and will compute a 2-factor in the current graph. This process takes $O(n^{\frac{3}{2}})$ in the worse case, so the contraction phase of the algorithm runs in $O(n^{\frac{5}{2}})$. The expansion phase of the algorithm takes $O(n)$ time in the worst case, since there are at most $O(n)$ expansions and each one is performed in constant time. Then, BIGCYCLE finds a 2-factor with at most $\frac{n}{7}$ cycles in $O(n^{\frac{5}{2}})$. ◀

We can now restate our main theorem from Section 1.1:

▶ **Theorem 1.** *Given a cubic bipartite connected graph G with n vertices, there is a polynomial time algorithm that computes a spanning Eulerian multigraph H in G with at most $\frac{9}{7}n$ edges.*

Proof. Theorem 13 proves that the COMPRESS and EXPAND phases of BIGCYCLE produce a 2-factor with at most $\frac{n}{7}$ cycles for the required class of graphs. Proposition 3 demonstrates that the DOUBLETREE phase BIGCYCLE successfully extends the 2-factor into a spanning Eulerian multigraph with at most $\frac{9}{7}n - 2$ edges.

From Theorem 13 that we can compute the required 2-factor in $O(n^{\frac{5}{2}})$. Once we have done this, we can compute the doubled spanning tree in $O(n)$ as well, as the graph has $O(n)$ edges. The total running time of the algorithm is $O(n^{\frac{5}{2}})$ in the worst case. ◀

Acknowledgment. We would like to thank Satoru Iwata and Alantha Newman for useful discussions during this project.

References

1. Nishita Aggarwal, Naveen Garg, and Swati Gupta. A 4/3-approximation for TSP on cubic 3-edge-connected graphs. *arXiv:1101.5586*, 2011.
2. David W Barnette. Conjecture 5. *Recent Progress in Combinatorics*, 343, 1969.
3. Sylvia Boyd, René Sitters, Suzanne van der Ster, and Leen Stougie. TSP on cubic and subcubic graphs. In *Integer Programming and Combinatoral Optimization*, pages 65–77. Springer, 2011.
4. Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, GSIA, Carnegie Mellon University, 1976.
5. José R Correa, Omar Larré, and José A Soto. TSP Tours in Cubic Graphs: Beyond 4/3. In *Algorithms–ESA 2012*, pages 790–801. Springer, 2012.
6. José R Correa, Omar Larré, and José A Soto. TSP Tours in Cubic Graphs: Beyond 4/3. *arXiv:1310.1896*, October 2013.
7. George Dantzig, Ray Fulkerson, and Selmer Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, pages 393–410, 1954.
8. Uri Feige, R Ravi, and Mohit Singh. Short tours through large linear forests. In *Integer Programming and Combinatoral Optimization*, pages 273–284. Springer, 2014.
9. David Gamarnik, Moshe Lewenstein, and Maxim Sviridenko. An improved upper bound for the TSP in cubic 3-edge-connected graphs. *Operations Research Letters*, 33(5):467–474, sep 2005.
10. Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. A randomized rounding approach to the traveling salesman problem. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 550–559. IEEE, 2011.

11 Michel X Goemans. Worst-case comparison of valid inequalities for the TSP. *Mathematical Programming*, 69(1-3):335–349, 1995.

12 Jeremy Karp and R. Ravi. A 9/7-Approximation Algorithm for Graphic TSP in Cubic Bipartite Graphs. *arXiv:1311.3640*, November 2013.

13 Tobias Mömke and Ola Svensson. Approximating graphic TSP by matchings. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 560–569. IEEE, 2011.

14 András Sebő and Jens Vygen. Shorter tours by nicer ears: 7/5-approximation for graphic tsp, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, 2014.

15 Nisheeth K Vishnoi. A permanent approach to the traveling salesman problem. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 76–80. IEEE, 2012.

Sherali-Adams Gaps, Flow-cover Inequalities and Generalized Configurations for Capacity-constrained Facility Location*

Stavros G. Kolliopoulos and Yannis Moysoglou

Department of Informatics and Telecommunications
National and Kapodistrian University of Athens
Panepistimiopolis Ilissia, Athens 157 84, Greece
{sgk,gmoys}@di.uoa.gr

Abstract

Metric facility location is a well-studied problem for which linear programming methods have been used with great success in deriving approximation algorithms. The capacity-constrained generalizations, such as capacitated facility location (CFL) and lower-bounded facility location (LBFL), have proved notorious as far as LP-based approximation is concerned: while there are local-search-based constant-factor approximations, there is no known linear relaxation with constant integrality gap. According to Williamson and Shmoys devising a relaxation-based approximation for CFL is among the top 10 open problems in approximation algorithms.

This paper advances significantly the state-of-the-art on the effectiveness of linear programming for capacity-constrained facility location through a host of impossibility results for both CFL and LBFL. We show that the relaxations obtained from the natural LP at $\Omega(n)$ levels of the Sherali-Adams hierarchy have an unbounded gap, partially answering an open question of [27, 6]. Here, n denotes the number of facilities in the instance. Building on the ideas for this result, we prove that the standard CFL relaxation enriched with the generalized flow-cover valid inequalities [1] has also an unbounded gap. This disproves a long-standing conjecture of [25]. We finally introduce the family of proper relaxations which generalizes to its logical extreme the classic star relaxation and captures general configuration-style LPs. We characterize the behavior of proper relaxations for CFL and LBFL through a sharp threshold phenomenon.

1998 ACM Subject Classification G.1.6 Optimization

Keywords and phrases Approximation Algorithms, Linear Programming, Facility Location

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.297

1 Introduction

Facility location is one of the most well-studied problems in combinatorial optimization. In the *uncapacitated* version (UFL) we are given a set F of facilities and set C of clients. We may open facility i by paying its opening cost f_i and we may assign client j to facility i by paying the connection cost c_{ij}. We are asked to open a subset $F' \subseteq F$ of the facilities and assign each client to an open facility. The goal is to minimize the total opening and connection cost. A *ρ-approximation algorithm,* $\rho \geq 1$, outputs in polynomial time a feasible

* This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) – Research Funding Program: "Thalis. Investing in knowledge society through the European Social Fund".

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 297–312

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

solution with cost at most ρ times the optimum. The approximability of general UFL is settled by an $O(\log |C|)$-approximation [18] which is asymptotically best possible, unless P = NP. In *metric* UFL the service costs satisfy the following variant of the triangle inequality: $c_{ij} \leq c_{ij'} + c_{i'j'} + c_{i'j}$ for any $i, i' \in F$ and $j, j' \in C$. This very natural special case of UFL is approximable within a constant-factor, and many improved results have been published over the years. In those, LP-based methods, such as filtering, randomized rounding and the primal-dual method have been particularly prominent (see, e.g., [33]). After a long series of papers the currently best approximation ratio for metric UFL is 1.488 [26], while the best known lower bound is 1.463, unless P = NP ([17] and Sviridenko [32]). In this paper we focus on two generalizations of metric UFL: the *capacitated facility location (*CFL*)* and the *lower-bounded facility location (*LBFL*)*.

CFL is the generalization of metric UFL where every facility i has a capacity u_i that specifies the maximum number of clients that may be assigned to i. In *uniform* CFL all facilities have the same capacity U. Finding an approximation algorithm for CFL that uses a linear programming lower bound, or even proving a constant integrality gap for an efficient LP relaxation, are notorious open problems. Intriguingly, the following rare phenomenon occurs. The natural LP relaxations have an unbounded integrality gap and the only known $O(1)$-approximation algorithms are based on local search, with the currently best ratios being 5 [9] for the non-uniform and 3 [4] for the uniform case respectively. In the special case where all facility costs are equal, CFL admits an LP-based approximation [25]. Comparing the LP optimum against the solution output by an LP-based algorithm establishes a guarantee that is at least as strong as the one established a priori by worst-case analysis. In contrast, when a local search algorithm terminates, it is not at all clear what the lower bound is. According to Williamson and Shmoys [33] devising a relaxation-based algorithm for CFL is one of the top 10 open problems in approximation algorithms.

A lot of effort has been devoted to understanding the quality of relaxations obtained by an iterative lift-and-project procedure. Such procedures define hierarchies of successively stronger relaxations, where valid inequalities are added at each level. After at most n levels, where n is the number of variables, all valid inequalities have been added and thus the integer polytope is expressed. Relevant methods include those developed by Balas et al. [8], Lovász and Schrijver [28] (for linear and semidefinite programs), Sherali and Adams [3], Lasserre [22] (for semidefinite programs). See [23] for a comparative discussion.

The seminal work of Arora et al. [7], studied integrality gaps of families of relaxations for Vertex Cover, including relaxations in the Lovász-Schrijver (LS) hierarchy. This paper introduced the use of hierarchies as a restricted model of computation for obtaining LP-based hardness of approximation results. Proving that the integrality gap for a problem remains large after many levels of a hierarchy is an unconditional guarantee against the class of relaxation-based algorithms obtainable through the specific method. At the same time, if an LP relaxation maintains a gap of g after a linear number of levels, one can take this as evidence that polynomially-sized relaxations are unlikely to yield approximations better than g (see also [29]). In fact, the former belief is now a theorem for maximum constraint satisfaction problems: in terms of approximation, LPs of size n^k, are exactly as powerful as $O(k)$-level Sherali-Adams relaxations [11].

LBFL is in a sense the opposite problem to CFL. In an LBFL instance every facility i comes with a lower bound b_i which is the minimum number of clients that must be assigned to i if we open it. In *uniform* LBFL all the lower bounds have the same value B. LBFL is even less well-understood than CFL. The first approximation algorithm for the uniform case had a performance guarantee of 448 [31], which has been improved to 82.6 [5]. Both use local search.

Apart from some work of the authors [21, 20] there has been no systematic theoretical study of the power of linear programming for approximating CFL. In [21] we show an unbounded gap for CFL at $\Omega(n)$ levels of the LS and the semidefinite mixed-LS$_+$ hierarchies, n being the number of facilities. In [20] we show that linear relaxations in the classic variables require at least an exponential number of constraints to achieve a bounded integrality gap. Note that it is well-known that hierarchies may produce an exponential number of inequalities already after one round. For related problems there are some recent interesting results. Improved approximations were given for k-median [27] and capacitated k-center [14, 6], problems closely related to facility location. For both, the improvements are obtained by LP-based techniques that include preprocessing of the instance in order to defeat the known integrality gap. For k-median, the authors of [27] state that their $(1+\sqrt{3}+\epsilon)$-approximation algorithm can be converted to a rounding algorithm on an $O(\frac{1}{\epsilon^2})$-level LP in the Sherali-Adams (SA) lift-and-project hierarchy. They propose exploring the direction of using SA for approximating CFL. In [6] the authors raise as an important question to understand the power of lift-and-project methods for capacitated location problems, including whether they automatically capture relevant preprocessing steps.

Our results. We give impossibility results on arguably the most promising directions for strengthening linear relaxations for CFL and LBFL and in doing so we answer open problems from the literature. Our contribution is threefold.

First, we show that the LPs obtained from the natural relaxations for CFL and LBFL at $\Omega(n)$ levels of the SA hierarchy have an unbounded gap on an instance where $|F| = \Theta(n)$ and $|C| = \Theta(n^3)$. This result answers the questions of [27] and [6] stated above as far as the natural LP is concerned and moreover it is asymptotically tight. In the instances we consider clients have unit demands and it is well known that in this case the integer polytope and the mixed-integer (where fractional client assignments are allowed) polytope are the same. Since SA extends to mixed-integer programs as well [13, 8], the mixed-integer polytope is obtained after at most n levels. Thus at most that many levels are needed also by the stronger, full-integer, SA procedure we employ, which in the lifting stage multiplies also with assignment variables. From a qualitative aspect, we give the first, to our knowledge, SA bounds for a relaxation where variables have more than one type of semantics, namely the facility opening and the client assignment type. Compare this, for example, with the Knapsack and Max Cut LPs that contain each one type of variable. The lifting of the assignment variables raises obstacles in the proof that we managed to overcome as discussed in Section 3.

We use the *local-to-global* method which was implicit in [7] for local-constraint relaxations and was then extended to the SA hierarchy in [15]. See also [16] for an explicit description and [12] for applications to Max Cut and other problems. In this approach, the feasibility of a solution for the t-level SA relaxation is established through the design of a set of appropriate distributions over feasible integer solutions for each constraint such that these global distributions agree with each other locally on relevant events. To prove Theorem 4 for CFL we devise first in Lemma 3 an intuitive method to construct an initial set of distributions for a constraint. These initial distributions are inadequate for constraints where all facilities appear as indices. An alteration procedure, explained in Propositions 3.1–3.3, produces the final set of distributions. Theorem 4 extends significantly our earlier result on the LS hierarchy for CFL [21] to the stronger SA hierarchy. It turns out that in both cases we can start from the same bad instance. It should be noted that the methodology in the two proofs is completely different – in [21] the result was obtained via an inductive construction of protection matrices.

Our second contribution (cf. Theorem 6) is that the *effective capacity* inequalities introduced in [1, 2] for CFL fail to reduce the gap of the classic relaxation to constant. These constraints generalize the flow-cover inequalities for CFL. Thus we disprove the long-standing conjecture of [25] that the addition of the latter to the classic LP suffices for a constant integrality gap. Our proof deviates from standard integrality gap constructions by applying the local-global method. The bad solution fools every inequality π because its part that is "visible" to π can be extended to a solution s^π that is a convex combination of feasible integer solutions. Our ideas can be extended to even more general families such as the *submodular inequalities* [1], cf. Theorem 7. All results in this paper make no time-complexity assumptions. To our knowledge no efficient separation algorithm for the effective capacity inequalities is known.

We finally introduce the family of proper relaxations which are configuration-like linear programs. The so-called *Configuration LP* was used by Bansal and Sviridenko [10] for the Santa Claus problem and has yielded valuable insights, mostly for resource allocation and scheduling problems (e.g., [30]). The analogue of the Configuration LP for facility location already exists, it is the *star relaxation* (see, e.g., [19]). We take the idea of a star to its logical extreme by introducing classes. A *class* consists of a set with an arbitrary number of facilities and clients together with an assignment of each client to a facility in the set. A *proper relaxation* for an instance is defined by a collection $\mathcal{C}$ of classes and a decision variable for every class. We allow great freedom in defining $\mathcal{C}$: the only requirement is that the resulting formulation is symmetric and valid. The *complexity* α of a proper relaxation is the maximum fraction of the available facilities that are contained in a class of $\mathcal{C}$. In Theorem 12 we characterize the behavior of proper relaxations for CFL and LBFL through a threshold result: anything less than maximum complexity results in unboundedness of the integrality gap, while there are proper relaxations of maximum complexity with a gap of 1.

Our results disqualify the so far most promising approaches for an efficient LP relaxation for CFL. Moreover, we advance drastically the state-of-the-art for the little understood LBFL. Whether a fundamentally new approach may succeed for either problem remains as an open question.

2 Preliminaries

Given an instance $I(F, C)$ of CFL or LBFL, we use n, m to denote $|F|$ and $|C|$ respectively. We will show our negative results for uniform, integer, capacities and lower bounds. Each client can be thought of as representing one unit of demand. It is well-known that in such a setting the splittable and unsplittable versions of the problem are equivalent. The following 0-1 IP is the standard valid formulation of uncapacitated facility location with unsplittable unit demands.

$$\min\{\textstyle\sum_{i\in F} f_i y_i + \sum_{i\in F}\sum_{j\in C} x_{ij}c_{ij} \;\mid\; x_{ij} \le y_i \;\; \forall i \in F, \forall j \in C,\\ \sum_{i\in F} x_{ij} = 1 \;\; \forall j \in C, \qquad y_i, x_{ij} \in \{0,1\} \;\; \forall i \in F, \forall j \in C\}$$

The linear relaxation results from the above IP by replacing the integrality constraints with: $0 \le y_i \le 1,\ 0 \le x_{ij} \le 1,\ \forall i \in F, \forall j \in C$. To obtain the standard LP relaxations for uniform CFL (and LBFL) with capacity U (lower bound B) the following constraints are added respectively:

$$\textstyle\sum_j x_{ij} \le U y_i \;\; \forall i \in F \quad \text{and} \quad \sum_j x_{ij} \ge B y_i \;\; \forall i \in F.$$

We will slightly abuse terminology by using the term *(LP-classic)* for both LPs. It will be clear from the context to which problem, CFL or LBFL, we refer.

We proceed to define the Sherali-Adams hierarchy [3]. Consider a polytope $P \subseteq \mathbb{R}^d$ defined by the linear constraints $Ax - b \leq 0$, $0 \leq x_i \leq 1$, $i = 1, \ldots, d$. We define the polytope $\mathrm{SA}^k(P) \subseteq \mathbb{R}^d$ as follows. For every constraint $\pi(x) \leq 0$ of P, for every set of variables $U \subseteq \{x_i \mid i = 1, \ldots, d\}$ such that $|U| \leq k$, and for every $W \subseteq U$, consider the valid constraint: $\pi(x) \prod_{x_i \in U - W} x_i \prod_{x_i \in W}(1 - x_i) \leq 0$. Linearize the system obtained this way by replacing (i) x_i^2 with x_i for all i and (ii) $\prod_{x_i \in I} x_i$ with x_I for each set $I \subseteq \{x_i | i = 1, \ldots, d\}$. $\mathrm{SA}^k(P)$ is the projection of the resulting linear system onto the singleton variables. We call $\mathrm{SA}^k(P)$ the polytope *obtained from P at level k* of the SA hierarchy. Given a cost vector $c \in \mathbb{R}^d$, the *relaxation obtained from P at level k* of SA is $\min\{c^T x \mid x \in \mathrm{SA}^k(P)\}$.

3 Sherali-Adams Gap for CFL

Consider an instance of metric Cfl with a total of $2n$ facilities, n with opening cost 0 which we call cheap (and denote the corresponding set by *Cheap*) and n with opening cost 1 which we call costly (and denote by *Costly*). The capacity $U = n^3$ and we have a total of $nU + 1$ clients. All connection costs are 0. We will show that the following bad solution s to the instance[1] survives a number of SA levels, which is linear in the number $2n$ of facilities, more specifically for $n/10$ levels. On the other hand, it is known that at level $2n$ the relaxation obtained expresses the integral polytope. Let $\alpha = n^{-2}$. For all $i \in Cheap$ and for all $j \in C$, $y_i = 1$ and $x_{ij} = \frac{1-\alpha}{n}$, and for all $i \in Costly$ and for all $j \in C$ $y_i = \frac{10}{n^2}$ and $x_{ij} = \frac{\alpha}{n}$. Theorem 4 below indicates that, as often with hierarchies, simple valid inequalities are generated after many rounds. The reader who is further interested in the robustness of SA for Cfl may consult Section 3.2.

The following lemma, which is implicit in previous work [15, 16] gives sufficient conditions for a solution to be feasible at level k of the SA hierarchy.

▶ **Lemma 1** ([15, 16]). *Let s be a feasible solution to the relaxation and let $v(\pi, z)$ be the set of variables appearing in a lifted constraint obtained from π multiplied by z. Solution s survives k levels of SA if for every constraint π and each multiplier z with at most k distinct variables there is:*

1. *A solution $s' = s_{\pi,z}$ which agrees with s on $v(\pi, z)$ such that s' is a convex combination E_d of integer solutions (and thus E_d defines a distribution on integer solutions) and*
2. *For any two sets $v(\pi_1, z_1)$ and $v(\pi_2, z_2)$, let $x_1 x_2 \cdot \ldots \cdot x_l$, $l \leq k+1$, be a product appearing in both lifted constraints obtained from π_1 and π_2 multiplied with z_1 and z_2 respectively. Then the probability $P[x_1 = 1 \wedge x_2 = 1 \wedge \ldots \wedge x_l = 1]$ is the same in both distributions E_{d_1} and E_{d_2} associated with $v(\pi_1, z_1)$ and $v(\pi_2, z_2)$ respectively.*

First consider a constraint $\pi\colon \sum_j x_{i^\pi j} \leq U y_{i^\pi}$ and a multiplier z. After multiplying by z and expanding, we obtain a linear combination of monomials (products). Then, for the $k < n - 1$ levels we consider there must be some costly facility $i_b \notin v(\pi, z)$. We construct a solution $s_{\pi,z} = (y', x')$ by setting $y'_{i_b} = 1 - \sum_{i \in Costly - \{i_b\}} y_i$ and letting all other variables the same as in the original bad solution s. We say that facility i_b *takes the blame*. We will prove that $s_{\pi,z}$ can be obtained as a convex combination E_d of a set of integer solutions satisfying constraint $\sum_{i \in Costly} y_i = 1$. While $s_{\pi,z}$ can be obtained as a convex combination

[1] The reader should notice that any similarity with Knapsack is superficial. Theorem 4 is about the Cfl polytope. Moreover, it is easy to embed our instance in a slightly larger one, with a non-trivial metric, so that the projection of the bad Cfl solution to the y-variables, is in the integral polytope of the "underlying" knapsack instance.

E_d in a variety of ways, we require that the assignments of clients to the cheap facilities are indistinguishable in E_d and the same must be true for the assignments to costly facilities other than i_b. In the upcoming definition, we use the product $p = z_1 z_2 \dots z_l$ as an abbreviation of the event $\mathcal{E}_p := \bigwedge_{i=1}^{l}(z_i = 1)$.

▶ **Definition 2.** Let i_b be the facility that takes the blame. We say that a distribution E_d is *assignment-symmetric* if the following are true:

1. $P_{E_d}[x_{i_{a_1} j_{b_1}} \dots x_{i_{a_t} j_{b_t}} y_{i_{a_{t+1}}} \dots y_{i_{a_l}}]$, with $t + l \leq k + 1$ is the same if we exchange all occurrences of cheap facility i_r by cheap facility $i_{r'}$ (in other words relabeling facilities). Note that we allow repetitions of facilities and clients in the description of the event.
2. $P_{E_d}[x_{i_{a_1} j_{b_1}} \dots x_{i_{a_t} j_{b_t}} y_{i_{a_{t+1}}} \dots y_{i_{a_l}}]$ is the same if we exchange all occurrences of client j_q by client $j_{q'}$.
3. $P_{E_d}[x_{i_{a_1} j_{b_1}} \dots x_{i_{a_t} j_{b_t}} y_{i_{a_{t+1}}} \dots y_{i_{a_l}}]$ is the same if we exchange all occurrences of costly facility i_1 by costly facility i_2, $i_1, i_2 \neq i_b$.

We can always obtain $s_{\pi,z}$ from such an assignment-symmetric distribution E_d as shown in the following lemma.

▶ **Lemma 3.** *Solution $s_{\pi,z}$ is a convex combination E_d of integer solutions which defines an assignment-symmetric distribution.*

Proof. We describe a probabilistic experiment which induces an assignment-symmetric distribution E_d over integer solutions satisfying $\sum_{i \in Costly} y_i = 1$.

Fix costly facility i_b. Let $w^1_{i_b} = \frac{\sum_j x'_{i_b j}}{y'_{i_b}}$ be the desired number of clients assigned to facility i_b in the integer solutions in E_d where facility i_b is opened. To simplify the presentation let us assume that $w^1_{i_b}$ and the w values we subsequently define are integers (we discuss later how to handle fractional w's). Let $w^1_{i_{ch}} = \frac{|C| - w_{i_b}}{|Cheap|}$ be the number of clients assigned to facility $i_c, c \in Cheap$. Likewise, fix costly facility $i_{co} \neq i_b$. Let $w^2_{i_{co}} = \frac{\sum_j x'_{i_{co} j}}{y'_{i_{co}}}$ be the number of clients assigned to facility i_{co} in each integer solution in E_d where facility i_{co} is opened and similarly let $w^2_{i_{ch}} = \frac{|C| - w_{i_{co}}}{|Cheap|}$ be the number of clients assigned to facility $i_c, c \in Cheap$, in each integer solution in E_d where facility i_{co} is opened. Observe that all the defined w's are less than U. The following procedure produces the assignment-symmetric distribution E_d.

Pick costly facility i_c with probability y'_{i_c}. If $i_c = i_b$ ($i_c \neq i_b$) then consider n bins corresponding to the n cheap facilities each one having w^1_{ch} (w^2_{ch}) slots and 1 bin corresponding to i_{co} having $w^1_{i_b}$ (w^2_{co}) slots. Randomly distribute $|C|$ balls to the slots of the $n + 1$ bins, with exactly one ball in each slot. Note that the above experiment induces a distribution over feasible integer solutions satisfying $\sum_{i \in Costly} y_i = 1$ since all the defined bin capacities are less than U and every client is assigned to exactly one opened facility in each outcome and exactly 1 costly facility is opened. Moreover the induced distribution E_d is assignment-symmetric and the expected (y, x) vector with respect to E_d is solution $s_{\pi,z}$.

Clearly, $s_{\pi,z}$ is the convex combination induced by E_d and E_d is assignment-symmetric: the cheap facilities are always open, and the costly are open a fraction of the time that is equal to the value of their corresponding y variable. The expected demand assigned to each $i_{co} \in Costly$ is $y'_{i_{co}} w_{co}$ which is the total demand assigned to i_{co} by $s_{\pi,z}$. Since the clients have the same probability of being tossed in the bin corresponding to i_{co}, the expected assignment of each client j to i_{co} is the same as in $s_{\pi,z}$.

As for the assignments to the cheap facilities, observe that in every outcome of the experiment the demand not assigned to costly facilities is exactly the demand assigned to cheap. Since we have proved that the expected assignments to the costly facilities are those

of the bad solution, by linearity of expectation we get that the total assignments to all cheap facilities are $\sum_{i\in Cheap}\sum_j x_{ij}$ (the total assignment of each client add up to 1 by the constraints of the LP). By the symmetric way the cheap are handled in the experiment we have that the total expected demand assigned to each $i \in Cheap$ is $\sum_j x_{ij}$ and by the symmetric way the clients are assigned to i through the experiment we get that the expected assignment of each j to i is x_{ij}.

To handle the case where the w's are not integers (which is actually always the case), we do the following: each time costly facility i_b ($i_c \neq i_b$) is picked, we set the number of slots of the corresponding bin to $\lfloor w^1_{i_b}\rfloor$ ($\lfloor w^2_{i_{co}}\rfloor$) with probability $1-(w^1_{i_b}-\lfloor w^1_{i_b}\rfloor)$ $(1-(w^2_{i_{co}}-\lfloor w^2_{i_{co}}\rfloor))$, otherwise set the slots to $\lceil w^1_{i_b}\rceil(\lceil w^2_{i_{co}}\rceil)$; this ensures that the expected number of slots is $w^1_{i_b}$ ($w^2_{i_{co}}$). The same rationale applies to the remaining cases of the construction. If the number of slots of i_b (i_{co}) is set to $\lfloor w^1_{i_b}\rfloor$ ($\lfloor w^2_{i_{co}}\rfloor$) then we pick some $n(\frac{|C|-\lfloor w^1_{i_b}\rfloor}{n}-\lfloor(\frac{|C|-\lfloor w^1_{i_b}\rfloor}{n})\rfloor)$ ($n(\frac{|C|-\lfloor w^2_{i_{co}}\rfloor}{n}-\lfloor(\frac{|C|-\lfloor w^2_{i_{co}}\rfloor}{n})\rfloor)$) cheap facilities at random and set their corresponding number of slots to $\lceil\frac{|C|-\lfloor w^1_{i_b}\rfloor}{n}\rceil$ ($\lceil\frac{|C|-\lfloor w^2_{i_{co}}\rfloor}{n}\rceil$) and the number of slots of the rest of the cheap facilities to $\lfloor\frac{|C|-\lfloor w^1_{i_b}\rfloor}{n}\rfloor(\lfloor\frac{|C|-\lfloor w^2_{i_{co}}\rfloor}{n}\rfloor)$. Otherwise pick some $n(\frac{|C|-\lceil w^1_{i_b}\rceil}{n}-\lfloor(\frac{|C|-\lceil w^1_{i_b}\rceil}{n})\rfloor)$ ($n(\frac{|C|-\lceil w^2_{i_{co}}\rceil}{n}-\lfloor(\frac{|C|-\lceil w^2_{i_{co}}\rceil}{n})\rfloor)$) cheap facilities at random and set their corresponding number of slots to $\lceil\frac{|C|-\lceil w^1_{i_b}\rceil}{n}\rceil$ ($\lceil\frac{|C|-\lceil w^2_{i_{co}}\rceil}{n}\rceil$) and the number of slots of the rest to $\lfloor\frac{|C|-\lceil w^1_{i_b}\rceil}{n}\rfloor(\lfloor\frac{|C|-\lceil w^2_{i_{co}}\rceil}{n}\rfloor)$. Note than in every case the expected number of slots per facility is as in the initial description of the experiment where we assumed the w values to be integers. ◀

We set the product-variables x_I appearing in constraint π multiplied by multiplier z to $P_{E_d}[I]$. Constraints $x_{ij}\leq y_i$, $x_{ij}\leq 1$, $y_i\leq 1$, are handled in the exact same way; the set of variables appearing in them is a subset of those appearing in the more complex constraints.

The second and more challenging case is when constraint π is $\sum_i x_{ij^\pi}=1$ for some client j^π. Let again z be a multiplier of level k. Observe now that all facilities in F appear in $v(\pi,z)$ as indexes of at least the x_{ij} variables. We select one facility i_b not appearing in z to *take the blame*. Let $s_{\pi,z}=(y',x')$ be the corresponding extended solution that can be written as a convex combination/assignment–symmetric distribution E_d of integer solutions; the existence of E_d is ensured by Lemma 3. In this case there is a major obstacle to the agreement of the products x_I: conditioning on the event x_{i_bj} the probability of an event $x_{ij'}, i\in Cheap$ for some $j'\neq j$ is higher than it would be if we were to condition on the event $x_{i'j}, i'\in Costly-\{i_b\}$. The same is true for more complex events involving assignments to cheap facilities conditioning on an assignment of facility i_b compared to the analogous event conditioning on some other costly facility. This can be problematic since facility i_b takes the blame in some distributions but does not in some others and thus there is the danger of violating the consistency required by the 2nd condition of Lemma 1. We overcome this difficulty by making alterations to E_d and constructing a distribution E_f where the probabilities of the aforementioned events are the same.

We now devise the altered distribution E_f. We first display the intuition in the following example: consider the event A: $x_{i_bj}=1\wedge x_{i_{ch}j'}=1$ and the event B: $x_{i_{co}j}=1\wedge x_{i_{ch}j'}=1$ with $i_{co}\in Costly-\{i_b\}$ and $i_{ch}\in Cheap$. The probability of A is $P[A]=P[x_{i_bj}=1]P[x_{i_{ch}j'}=1\mid x_{i_bj}=1]=x'_{i_bj}\frac{w^1_{ch}}{|C|-1}$ and the probability of B is $P[B]=P[x_{i_{co}j}=1]P[x_{i_{ch}j'}=1\mid x_{i_{co}j}=1]=x'_{i_{co}j}\frac{w^2_{ch}}{|C|-1}$. Note that $P[A]\approx P[B](1+1/n)$ so $P[A]$ is only slightly greater. We nullify the difference between those probabilities by performing an alteration step to distribution E_d that we call *transfusion of probability*. We pick some

measure of an integer solution s_1 for which $x_{i_{ch}j'} = 1 \wedge x_{i_bj} = 1 \wedge x_{i_bj''} = 0$ for some client j''. We pick the same quantity of measure of some integer solution (or of some set of solutions) s_2 for which $x_{i_{ch}j'} = 0 \wedge x_{i_bj} = 0 \wedge x_{i_bj''} = 1$ and we exchange the values of the assignments $x_{i_bj}, x_{i_bj''}$ of the solutions. Let that quantity be $P[A] - P[B]$, it is easy to see that each set of solutions has enough measure to perform the transfusion. The resulting distribution E_f now has $P[A] = P[B]$. In general, when transfusing probabilistic measure for complex events, we must be careful not to change the probability of events involving only assignments to cheap facilities, as opposed to the simplified example above.

Now let p be a product appearing in constraint π after having multiplied by multiplier z. We only consider products where exactly one variable x_{i_bj} appears. Recall we chose i_b so that it does not appear in z; thus we cannot have y_{i_b} or more than one assignments of i_b appearing in a product p. We may also assume that there is no y_i variable in p, since if there is for some $i \in Costly - \{i_b\}$ the probability of $\mathcal{E}_p$ is simply 0 and if $i \in Cheap$ the we can ignore the effect of $y_i = 1$ since it is always true. Likewise we assume that there is no assignment variable of another costly facility. We shall make corrections of the probability of all such events $\mathcal{E}_p$ in a top-down manner: at step i we fix the probability of all the events $x_{i_bj} = 1 \wedge x_{i_{a_1}j_{b_1}} = 1 \wedge \ldots \wedge x_{i_{a_{k-i+1}}j_{b_{k-i+1}}} = 1$ where $x_{i_bj}x_{i_{a_1}j_{b_1}} \ldots x_{i_{a_{k-i+1}}j_{b_{k-i+1}}}$ is a product p appearing in constraint π multiplied by z. In other words, we fix the probabilities in decreasing order of the cardinality of the set of variables appearing in p. The following proposition relates the probability of $\mathcal{E}_p$ with that of $\mathcal{E}_{p'} = \mathcal{E}_{px_{ij}}$, an event with the additional requirement that $x_{ij} = 1$.

▶ **Proposition 3.1.** *Let $p = x_{i_bj}x_{i_{a_1}j_{b_1}}x_{i_{a_2}j_{b_2}} \ldots x_{i_{a_l}j_{b_l}}$ and let $p' = px_{i_{a_{l+1}}j_{b_{l+1}}}$. Then in E_d, $(1 - o(1))P[\mathcal{E}_p]/n \leq P[\mathcal{E}_{p'}] \leq (1 + o(1))P[\mathcal{E}_p]/n$.*

Consider step i of the above iterative construction of E_f. Let $p = x_{i_bj}x_{i_{a_1}j_{b_1}} \ldots x_{i_{a_{k-i+1}}j_{b_{k-i+1}}}$ and the event $\mathcal{E}_p \colon x_{i_bj} = 1 \wedge x_{i_{a_1}j_{b_1}} = 1 \wedge \ldots \wedge x_{i_{a_{k-i+1}}j_{b_{k-i+1}}} = 1$. We wish in E_f the probability $P[\mathcal{E}_p]$ to be equal to $P[\mathcal{E}_{p/fixed}] = P[x_{i^*j} = 1 \wedge x_{i_{a_1}j_{b_1}} = 1 \wedge \ldots \wedge x_{i_{a_{k-i+1}}j_{b_{k-i+1}}} = 1]$ in E_d for $i^* \in Costly - \{i_b\}$. We bound the ratio $\frac{P[\mathcal{E}_p]}{P[\mathcal{E}_{p/fixed}]}$:

▶ **Proposition 3.2.** *Let $\mathcal{E}_p$ and $\mathcal{E}_{p/fixed}$ be defined as above. Then $(1 + (1 - o(1))1/n)^{k-i+1} \leq \frac{P[\mathcal{E}_p]}{P[\mathcal{E}_{p/fixed}]} \leq (1 + (1 + o(1))1/n)^{k-i+1}$.*

Now we describe in detail the alterations of the probabilities in each iteration. The corrections of the probabilities of events of previous iterations affect the probabilities of the events of the current iteration of the procedure that constructs E_f. We bound this effect on the probability of an event $\mathcal{E}_p$ of the current iteration i by considering the corrections of the events $\mathcal{E}_{p'} = \mathcal{E}_p \wedge x_{ij} = 1$, with x_{ij} in the set of variables appearing in z and $x_{ij} \notin \mathcal{E}_p$, of the previous iteration and using the union bound.[2] There are exactly i events needed to be taken into consideration for each such $\mathcal{E}_p$ of the current step i. The amount of the effect of the correction of the previous iteration is by Proposition 3.2 at most $i((1+(1+o(1))1/n)^{k-i+2} - 1)P[\mathcal{E}_{p'/fixed}]$ while the measure of the needed correction for $\mathcal{E}_p$ is at least $((1+(1-o(1))1/n)^{k-i+1} - 1)P[\mathcal{E}_{p/fixed}]$ which by Proposition 3.1 and by the number of rounds we consider is higher, in particular $((1 + (1 - o(1))1/n)^{k-i+1} - 1)P[\mathcal{E}_{p/fixed}] \geq n(1-o(1))((1+(1-o(1))1/n)^{k-i+1}-1)P[\mathcal{E}_{p'/fixed}] > i((1+(1+o(1))1/n)^{k-i+2}-1)P[\mathcal{E}_{p'/fixed}]$. To subtract from $P[\mathcal{E}_p]$ the rest of the probabilistic measure required from the correction at

[2] Notice that any effect of iteration $j < i - 1$ on $P[\mathcal{E}_p]$, originates from events that are subsets of $\mathcal{E}_{p'}$ and has therefore been accounted for.

step i, say a measure of μ, we do the following transfusion step: pick a measure μ of solutions from distribution E_d such that $x_{i_b j} = 0$, $x_{i_b j'} = 1$ for any j' that does not appear as index of any variable in $v(\pi, z)$, all the other events of $\mathcal{E}_p$ are false, and so are all the remaining events corresponding to assignments in z. Then pick an equal measure of solutions from E_d such that $x_{i_b j} = 1$, $x_{i_b j'} = 0$, all the other events of $\mathcal{E}_p$ are true, and all the remaining events corresponding to assignments in z are false. Now exchange the values of the assignments of j and j' of the solutions of the two sets. The resulting distribution has the probability of $\mathcal{E}_p$ fixed to the desired value and moreover, by the choice of the sets of solutions on which we perform the transfusion step, the probability of the events fixed in previous iterations was not altered and neither was the probability of events containing only assignments of cheap facilities. Clearly, the solution $s_{\pi,z}$ is still obtained in expectation. It remains to show that the transfusion step can be performed, i.e., that there is enough measure μ in the involved sets of integer solutions.

▶ **Proposition 3.3.** *The probabilistic transfusion step of the above iterative procedure can always be performed.*

Proof. The intuition behind the proof is that the "donor" event that supplies the required measure is much more likely to occur than the events that require the transfusion.

Consider the measure t in E_d of the set of integer solutions satisfying $y_{i_b} = 1$ and all events encountered at any iteration being false, namely $x_{i_b j} = 0 \wedge x_{i_1 j_1} = 0 \wedge x_{i_2 j_2} = 0 \wedge \ldots \wedge x_{i_k j_k} = 0$. Then, by the random experiment of the construction of E_d, this event is equivalent to the event that facility i_b is picked, $x_{i_b j} = 0$ and the k balls corresponding to the clients of the rest of the events are not tossed in their corresponding bins. Using again that both w^1_{ch}, w^2_{ch} are $\Theta(n^3)$ and $k < n$, we can bound the probability of the k balls by that of k Bernoulli trials with probability of success $2/n$ (we are once again very generous). Then the probability that all events fail is at least $(1 - 2/n)^k > \lim_{n\to\infty}(1 - 2/n)^n = 1/e^2$. Thus measure t is at least $(y_{i_b} - x_{i_b j})1/e^2$ which is constant. On the other hand the measure required by the transfusion step for each event $\mathcal{E}_p$ of iteration i that needs to be fixed is at most $(e^2 - 1)P[\mathcal{E}_{p/fixed}] = \Theta(1/n^i)$. There are $\binom{k+1}{k-i+1}$ such events of iteration i, and summing over all the iterations of our construction we get $\sum_{i=1}^{k} \binom{k+1}{k-i+1}\Theta(1/n^i)$ which quantity is less than $(y_{i_b} - x_{i_b j})1/e^2$ for the $k = n/10$ levels of SA we consider, so we can always pick the required amount of measure. ◀

▶ **Theorem 4.** *There is a family of* Cfl *instances with* $2n$ *facilities and* $n^4 + 1$ *clients such that the relaxations obtained from (LP-classic) at* $\Omega(n)$ *levels of the Sherali-Adams hierarchy have an integrality gap of* $\Omega(n)$.

Proof. For each lifted constraint π multiplied by multiplier z at level t, the corresponding distribution E_d or E_f is clearly a distribution over integer solutions, so the first condition of Lemma 1 is satisfied. For the second condition, observe that if an event $\mathcal{E}_p$ involves more than one costly facility, it has 0 probability in all distributions. If an event $\mathcal{E}_p$ involves only cheap facilities, it has the same probability in all distributions E_f and E_d, since in the construction of a distribution E_f we took care not to change the probability of such events. An event $\mathcal{E}_p$ that involves more than one assignment of a costly facility (but no other costly) has in every distribution E_f the same probability (which is the same as in every E_d) since in the construction of E_f we did not alter the probabilities of such events. And lastly, when an event $\mathcal{E}_p$ involves exactly one assignment of some costly facility i_x, note that in some cases i_x takes the blame but in other cases it does not, depending on $v(\pi, z)$. But due to the iterative procedure of probabilistic transfusion, the probability of event $\mathcal{E}_p$ in a distribution in which

i_x is not the facility that takes the blame is equal to the probability of the same event in the distributions that i_x takes the blame. So Lemma 1 holds. It is easy to see that bad solution has cost $\Theta(n^{-1})$ while any feasible solution to the instance has cost $\Omega(1)$. ◀

3.1 SA Gap for LBFL

A similar result to Theorem 4 can be proved for LBFL. Consider an instance with n facilities, lower bound $B = n^3$ and a total of $n(B-1)$ clients. The metric space here is more intriguing than the one for the CFL case. Consider a regular $(n-1)$-dimensional simplex with edge length 1. On each of the n vertices of the simplex a facility along with some $B-1$ clients are located. All opening costs are 0. Clearly every integer solution has a cost of at least $B-1$ since we can open at most $n-1$ of the facilities, and so at least $B-1$ clients will have to be assigned to some facility other than the one on the same vertex. We call a client j that is located on the same vertex with facility i, *exclusive* client of i. We denote by $Exclusive(i)$ the set of clients that are exclusive to facility i. On the other hand we can show that the following bad solution s is feasible at $\Omega(n)$ levels of the SA hierarchy. For all $i \in F$, $y_i = 1 - n^{-2}$; for a client $j \in C$, $x_{ij} = 1 - 10n^{-2}$, if $j \in Exclusive(i)$, and $x_{ij} = \frac{10n^{-2}}{n-1}$ for all other facilities. Solution s incurs a cost of $o(B)$.

▶ **Theorem 5.** *There is a family of* LBFL *instances with n facilities and $n^4 - n$ clients such that the relaxations obtained from (LP-classic) at $\Omega(n)$ levels of the Sherali-Adams hierarchy have an integrality gap of $\Omega(n)$.*

The proof is similar to that of CFL and is thus omitted. Here the reader can find a sketch of the necessary changes to the proof of Theorem 4.

Proof sketch of Theorem 5. Consider a constraint $\pi : \sum_j x_{i^\pi j} \geq B y_{i^\pi}$ and a multiplier z at level k and let $v(\pi, z)$ be the set of variables appearing in the multiplied constraint. We pick a facility i_b not in $v(\pi, z)$ to take the blame. We construct a solution s' where we set $y'_{i_b} = n - 1 - \sum_{i \neq i_b} y_i$ and for each $j \in Exclusive(i_b)$ we set $x'_{i_b j} = y'_{i_b} = \frac{1-1/n}{n}$ and we distribute the remaining demand that was assigned to i_b to each facility from a constant-size set I_b of facilities not appearing in $v(\pi, z)$. Solution s' can be obtained as a convex combination of integer solutions by constructing a distribution similarly to Lemma 3. This time the distribution satisfies that exactly $n-1$ facilities are opened in each outcome of the experiment. Note that we do not require the underlying distribution to be assignment symmetric, because facilities have to treat differently their exclusive clients. We set the values of the linearized products appearing in the multiplied constraint equal to the probability of the corresponding events with respect to the aforementioned distribution. No product involving variables of $i_b \cup I_b$ appear in the constraint. For constraints $0 \leq x_{ij}, y_i \leq 1$ and $x_{ij} \leq y_i$ the construction of the distribution is the same. The distributions constructed so far are locally consistent as required by Lemma 1.

The case where the constraint is $\pi : \sum_i x_{ij^\pi} = 1$ is once again more complicated. We choose a facility $i_b \notin z$ and moreover $j^\pi \notin Exclusive(i_b)$ to take the blame and the set I_b is defined as before except we also require that j^π is not exclusive to any of them. Solution s' is constructed like in the previous case. All products take the value of the corresponding events in the distribution except those in which the unique variable involving i_b appears, namely $x_{i_b j}$ and those involving facilities in I_b. We perform a transfusion step so that the probabilities of all the events whose corresponding products appear in the lifted constraint become consistent with the distributions of the previous case: this time we need to fix the probabilities of the events involving facility i_b or some facility $i \in I_b$. ◀

3.2 Robustness of the SA Gap

In this section we explain how adding simple valid inequalities does not affect our arguments on the SA hierarchy.

As an example we address the valid inequality $\sum_i y_i \geq \lceil D/U \rceil$, where D is the total amount of demand. This is a well-known facet-inducing constraint for our instance, see, e.g., [24, p. 283]. Of course this inequality is rendered useless by slight modifications to the instance and the bad solution. Identifying "areas" of a fractional solution where the demand exceeds the available capacity is impossible without some yet unknown form of preprocessing. In fact part of the motivation behind Theorem 4 is to demonstrate that the SA hierarcy is inadequate for such preprocessing purposes.

We modify the family of "bad" instances by using the same trick we used in the proof of Theorem 6: we have n cheap and n costly facilities and $Un+1$ clients, and the bad solution in which for every $ch \in Cheap$, $co \in Costly$, and client j, $y_{ch} = 1, x_{chj} = \frac{1-\alpha}{n}$, $y_{co} = 10/n^2, x_{coj} = \frac{\alpha}{n}$ with $\alpha = n^{-2}$, and additionally we add a set of n *dummy* facilities a_i, $1 \leq i \leq n$, all with 0 opening costs, on the same point at distance 1 from the rest. In the bad solution s we additionally set $y_{a_i} = 1$ and $x_{a_i j} = 0$ for all i and for all clients j. The inequality is obviously satisfied.

In the design of the locally consistent distributions, now we must give a distribution for the case where the constraint π is the new one $\sum_i y_i \geq \lceil D/U \rceil$, and verify that the "visible" part of the distribution agrees with the visible part of all other distributions of the proof. In this case there must be some dummy facility a_d not appearing as an index in the multiplier z of the constraint (although its y variable does appear in π). Additionally there must be a costly facility i' for which the assignments of clients to i' do not appear in $v(\pi, z)$ – this is ensured by the number of rounds we consider. We modify the solution (y, x) to obtain (y', x') where the facilities i' and a_d exchange the values of their corresponding assignments. We define now the random experiment similarly to the proof of Lemma 3 with facility a_d taking the blame. The only difference is that while a_d is opened 100% of the time, it is not assigned any demand when a costly facility other than i' is opened. In the terminology of Theorem 6 that follows, a_d is always open but it is inactive when some $i \in Costly$, $i \neq i'$, is opened. It is easy to see that the distribution obtained is consistent with all the other distributions defined for this modified instance, as required by Lemma 1.

4 Fooling the Effective Capacity Inequalities for CFL

In this section we show that the (LP-classic) for Cfl with the addition of the effective capacity inequalities proposed in [1] has unbounded gap.

Consider the general case where facility i has capacity u_i and client j has demand d_j. For a set J of clients, we denote their total demand by $d(J) = \sum_{j \in J} d_j$. Let $J \subseteq C$ be a set of clients, let $I \subseteq F$ be a set of facilities, and let $J_i \subseteq J$ be a set of clients for each facility $i \in I$. Given a facility i, we denote the *effective capacity* of i with respect to J_i by $\bar{u}_i = \min\{u_i, d(J_i)\}$. I is a *cover* with respect to J if $\sum_{i \in I} \bar{u}_i = d(J) + \lambda$ with $\lambda > 0$. λ is called the *excess capacity*. Let $(x)^+ = \max\{x, 0\}$. In the case where $J_i = J$ for all $i \in I$ the following inequalities called *flow-cover* inequalities were introduced for Cfl in [1].

$$\sum_{i \in I} \sum_{j \in J} d_j x_{ij} + \sum_{i \in I} (u_i - \lambda)^+ (1 - y_i) \leq d(J)$$

If $\max_{i \in I}(\bar{u_i}) > \lambda$, the following inequalities, called the *effective capacity inequalities* are valid and strengthen the flow-cover inequalities [1].

$$\sum_{i\in I}\sum_{j\in J_i} d_j x_{ij} + \sum_{i\in I}(\bar{u}_i - \lambda)^+(1-y_i) \le d(J)$$

The proof of the following theorem uses some of the ideas we introduced earlier for Theorem 4.

▶ **Theorem 6.** *The integrality gap of the relaxation obtained from (LP-classic) with the addition of the effective capacity inequalities is* $\Omega(n)$*, where* n *is the number of facilities in the instance.*

Proof. Consider an instance with n cheap and $n+2$ costly facilities and $Un+1$ clients, $U = n^3$. Define the bad solution s, similarly to Section 3, s.t. for every $ch \in Cheap$, $co \in Costly$, and client j, $y_{ch} = 1, x_{chj} = \frac{1-\alpha}{n}$, $y_{co} = 10/n^2, x_{coj} = \frac{\alpha}{n+2}$. Recall that $\alpha = n^{-2}$. We add a set of $n+2$ facilities a_i, $1 \le i \le n+2$, all with 0 opening costs, on the same point at distance 1 from the rest (an instance of the so-called *facility location on a line*). In the bad solution s we additionally set $y_{a_i} = 1$ and $x_{a_i j} = 0$ for all i and for all clients j.

We will prove that in every cover I with respect to some client set J and to the J_i client sets for each i, there must always be a number of at least $2n^3$ clients whose assignment variables to some costly and to some a_i do not appear in the constraint. This is because if, $\bar{u}_i = U$ for each $i \in Costly$, or, $\bar{u}_{a_i} = U$ for each $i \le n+2$, then the excess capacity $\lambda > U$ since $d(J) \le Un + 1$. This contradicts the requirement that $\lambda < U$. So there must be a costly facility $i_{co'}$ and some facility $a_{i'}$ such that for the corresponding sets we have $|J_{i_{co'}}|, |J_{a_{i'}}| < U$, and so there is a set J^* of $2n^3$ clients whose assignments to those two facilities do not appear in the constraint. We exchange the values of $x_{i_{co'}j}$ and $x_{a_{i'}j}$ for all $j \in J^*$, leaving everything else the same, and we obtain a solution $s' = (y', x')$. We can prove similarly to the proof of Lemma 3 that s' is a convex combination of integer solutions and thus solution s satisfies the inequality since the parts of s and s' visible to that inequality are the same.

We modify the construction of Lemma 3 in the following way: facility $a_{i'}$ is opened 100% of the time but is active $1 - \sum_{i\in Costly} y'_i$ of the time, when none of the costly facilities are opened. When it is not active, the capacity of its corresponding bin is 0. When a costly other than $i_{co'}$ is opened the experiment is the same as in Lemma 3. If costly facility $i_{co'}$ is opened the capacity of the corresponding bin is $w^2_{co'} = \frac{\sum_j x'_{co'j}}{y'_{i_{co'}}}$ and the capacity of the cheap is $\frac{|C| - w^2_{co'}}{n}$. We randomly select some $w^2_{co'}$ clients that do not belong to J^* to be tossed in the bin of $i_{co'}$; we randomly distribute the balls corresponding to the remaining clients to the slots of the cheap facilities. When $a_{i'}$ is active, and thus no costly facility is opened, the capacity of the corresponding bin is $w^1_{a_{i'}} = \frac{\sum_j x'_{a_{i'}j}}{1 - \sum_{i\in Costly} y'_i}$ and the capacity of the cheap is $\frac{|C| - w^1_{a_{i'}}}{n}$. We select randomly some $w^1_{a_{i'}}$ clients in J^* and we toss the corresponding balls in the bin of $a_{i'}$. We randomly toss the remaining balls to the slots of the bins of the cheap facilities.

Note that the above experiment induces a distribution over feasible integer solutions since all the defined bin capacities are less than U (this is by the choice of the size of J^*) and every client is assigned to exactly one opened facility in each outcome. We do not need this distribution to be assignment-symmetric. Observe that the expected vector with respect to the latter distribution is solution s'. Finally, note that we once again treated the capacities w of the bins as being integral. For fractional bin capacities (which is actually always the case for the defined w's) we can define the experiment in a similar way to the proof of Lemma 3. ◀

The submodular inequalities introduced in [1] are even stronger than the effective capacity inequalities. We limit our discussion to uniform CFL where all clients have unit demands.

Choose a subset $J \subseteq C$ of clients, and let $I \subseteq F$ be a subset of facilities. For each facility $i \in I$ choose a subset $J_i \subseteq J$. Consider a 3-level network G with a source s, a set of nodes corresponding to the facilities, a set of nodes corresponding to the clients and a sink t. The source s is connected by an edge of capacity $\min\{U, |J_i|\}$ to each facility node i. That node is connected by an edge of unit capacity to each node corresponding to client j, $j \in J_i$. Each node corresponding to some client is connected by an edge of unit capacity to the sink t.

Define $f(I)$ as the maximum s-t flow value in G. Define $f(I \setminus \{i\})$ as the maximum flow when facility i is closed, i.e., when the capacity of edge (s, i) is set to zero. The difference in maximum flow when all facilities in I are open, and when all facilities except facility i are open, is called the *increment* function and is defined as $\rho_i(I \setminus \{i\}) = f(I) - f(I \setminus \{i\})$.

For any choice of $I \subseteq F$, $J \subseteq C$, and $J_i \subseteq J$, for all i, the following inequalities, called *the submodular inequalities,* are valid [1]. The name reflects the fact that the function $f(I)$ is submodular.

$$\sum_{i \in I} \sum_{j \in J_i} x_{ij} + \sum_{i \in I} \rho_i(I \setminus \{i\})(1 - y_i) \leq f(I)$$

Theorem 7 below strictly generalizes Theorem 6 to the submodular inequalities.

▶ **Theorem 7.** *The integrality gap of the relaxation obtained from (LP-classic) with the addition of the submodular inequalities is* $\Omega(n)$*, where* n *is the number of facilities in the instance.*

5 Proper Relaxations

In this section we present the family of proper relaxations and characterize their strength. Consider a 0-1 (y, x) vector on the set of variables of the classic relaxation (LP-classic) such that $y_i \geq x_{ij}$ for all $i \in F, j \in C$. The meaning of $y_i = 1$ is the usual one that we open facility i. Likewise, the meaning of $x_{ij} = 1$ is that we assign client j to facility i. We call such a vector a *class*. Note that the definition is quite general and a class can be defined from any such (y, x), which may or may not have a relationship to a feasible integer solution. We denote the vector corresponding to a class cl as $(y, x)_{cl}$. We associate with class cl the *cost of the class* $c_{cl} = \sum_{i|y_i=1 \in (y,x)_{cl}} f_i + \sum_{i,j|x_{ij}=1 \in (y,x)_{cl}} c_{ij}$. Let the *assignments of class* cl be defined as $Agn_{cl} = \{(i, j) \in F \times C \mid x_{ij} = 1 \text{ in } (y, x)_{cl}\}$. We say that cl *contains* facility i, if the corresponding entry y_i in the vector $(y, x)_{cl}$ equals 1. The set of facilities contained in cl is denoted by $F(cl)$.

▶ **Definition 8** (Constellation LPs)**.** Let $\mathcal{C}$ be a set of classes defined for an instance $I(F, C)$ of CFL or LBFL. Let x_{cl} be a variable associated with class $cl \in \mathcal{C}$. The *constellation LP with class set* $\mathcal{C}$, denoted LP($\mathcal{C}$), is defined as $\min\{\sum_{cl \in \mathcal{C}} c_{cl} x_{cl} \mid \sum_{cl|\exists i:(i,j) \in Agn_{cl}} x_{cl} = 1 \ \forall j \in C, \quad \sum_{cl|i \in F(cl)} x_{cl} \leq 1 \ \forall i \in F, \quad x_{cl} \geq 0 \ \forall cl \in \mathcal{C}\}$.

We refer simply to a *constellation LP* when $\mathcal{C}$ is implied from the context. We define the *projection* $s' = (y^{s'}, x^{s'})$ of solution $s = (x^s_{cl})_{cl \in \mathcal{C}}$ of LP($\mathcal{C}$) to the facility opening and assignment variables (y, x) as $y_i^{s'} = \sum_{cl|i \in cl} x^s_{cl}$ and $x_{ij}^{s'} = \sum_{cl|(i,j) \in Agn_{cl}} x^s_{cl}$. We restrict our attention to constellation LPs that satisfy a symmetry property that is very natural for uniform capacities and unit demands.

▶ **Definition 9** (P_1: Symmetry)**.** We say that property P_1 holds for the constellation linear program LP($\mathcal{C}$) if for every class $cl \in \mathcal{C}$, all classes resulting from a permutation that relabels the facilities and/or the clients of cl are also in $\mathcal{C}$.

▶ **Definition 10** (Proper Relaxations)**.** We call *proper relaxation* for CFL (LBFL) a constellation LP that is valid and satisfies property P_1.

A simple example of a constellation LP is the well-known *(LP-star)* (see, e.g., [19]) where $\mathcal{C}$ corresponds to the set of all *stars*: a facility and a set of at most U (or at least B for LBFL) clients assigned to it. Obviously (LP-star) is a proper relaxation, while (LP-classic) is equivalent to (LP-star). Therefore proper relaxations generalize the known natural relaxations for CFL and LBFL. In order to characterize the strength of a proper LP we need the notion of complexity.

▶ **Definition 11** (Complexity of proper relaxations)**.** Given an instance $I(F, C)$ of CFL (LBFL) let F' be a maximum-cardinality set of open facilities in an integral feasible solution. The *complexity* α of a proper relaxation $LP(\mathcal{C})$ for I is defined as the $\sup_{cl\in\mathcal{C}}(|F(cl)|/|F'|)$.

The complexity of a proper LP represents the maximum fraction of the total number of feasibly openable facilities that is allowed in a single class. A complexity of nearly 1 means that there are classes that take each into consideration almost the whole instance at once. Low complexity means that all classes consider the assignments of a small fraction of the instance at a time.

▶ **Theorem 12.** *Every proper relaxation for uniform* CFL *(*LBFL*) with complexity $\alpha < 1$ has an unbounded integrality gap. There is a proper relaxation for* CFL *(*LBFL*) of complexity* 1 *whose projection to (y, x) expresses the integral polytope.*

Proof sketch of Theorem 12 for LBFL**.** We are given an arbitrary proper relaxation $LP(\mathcal{C})$ of complexity $\alpha < 1$, for an instance with $n + 1$ facilities, n^3 clients and $B = n^2$, and the following metric distances: put every facility i, $i \leq n - 1$, together with $B - 1$ clients, which we call *exclusive* clients of i, on a distinct vertex of an $(n - 2)$-dimensional regular simplex in $\mathbb{R}^{n-2}$ with edge length D. Put facilities $n, n + 1$ together with their exclusive clients, which are all the $B + n - 1$ remaining clients, to a point far away from the simplex, so the minimum distance from a vertex is $D' = \Omega(nD)$. We set all the facility costs to 0.

A major challenge is that we have no a priori knowledge of $\mathcal{C}$. We use the validity of $LP(\mathcal{C})$ and the fact that $\alpha < 1$, to prove that there is a class cl_0 with some desired properties that must belong to $\mathcal{C}$. Using classes that are symmetric to cl_0, which also must belong to $\mathcal{C}$, we construct a vector s that is feasible for $LP(\mathcal{C})$ and whose projection on the classic variables is the following (y^*, x^*): for each facility $i \leq n - 1$, its exclusive clients are assigned to it with a fraction of $\frac{n^2-1}{n^2}$ each, while they are assigned with a fraction of $\frac{1}{(n^2)(n-2)}$ to each other facility $i' \leq n - 1$. As for facilities $n, n + 1$, all of their exclusive clients are assigned with a fraction of 1/2 to each. Moreover $y^*_i = \frac{n^2-1}{n^2}$, for $i \leq n - 1$, and $y^*_n = y^*_{n+1} = \frac{n^2+n-1}{2n^2}$.

The cost of the fractional solution we constructed is $O(nD)$ due to the assignments of exclusive clients of facility i, $i \leq n - 1$, to facilities i' with $i' \neq i$, $i' \leq n - 1$. As for the cost of an arbitrary integral solution, observe that since the $n^2 + n - 1$ exclusive clients of $n, n + 1$ are very far from the rest of the facilities, using n of them to satisfy some demand of those facilities and help to open all of them, incurs a cost of $\Omega(nD') = \Omega(n^2 D)$. On the other hand, if we do not open all of the $n - 1$ facilities on the vertices of the simplex (since they have in total $(n - 1)(B - 1)$ exclusive clients which is not enough to open all of them), there must be at least one such facility not opened in the solution, thus its $B - 1 = \Theta(n^2)$ exclusive clients must be assigned elsewhere, incurring a cost of $\Omega(n^2 D)$. ◀

References

1 Karen Aardal, Yves Pochet, and Laurence A. Wolsey. Capacitated facility location: Valid inequalities and facets. *Mathematics of Operations Research*, 20:562–582, 1995.

2 Karen Aardal, Yves Pochet, and Laurence A. Wolsey. Erratum: Capacitated facility location: Valid inequalities and facets. *Mathematics of Operations Research*, 21:253–256, 1996.

3 Warren P. Adams and Hanif D. Sherali. Linearization strategies for a class of zero-one mixed integer programming problems. *Oper. Res.*, 38(2):217–226, April 1990.

4 Ankit Aggarwal, Anand Louis, Manisha Bansal, Naveen Garg, Neelima Gupta, Shubham Gupta, and Surabhi Jain. A 3-approximation algorithm for the facility location problem with uniform capacities. *Math. Program.*, 141(1-2):527–547, 2013.

5 Sara Ahmadian and Chaitanya Swamy. Improved approximation guarantees for lower-bounded facility location. In *Proceedings of the 10th International Workshop on Approximation and Online Algorithms* , WAOA'12, pages 257–271, Ljubljana, Slovenia, 2012.

6 Hyung-Chan An, Aditya Bhaskara, and Ola Svensson. Centrality of trees for capacitated k-center. *CoRR*, abs/1304.2983, 2013.

7 Sanjeev Arora, Béla Bollobás, László Lovász, and Iannis Tourlakis. Proving integrality gaps without knowing the linear program. *Theory of Computing*, 2(1):19–51, 2006.

8 Egon Balas, Sebastián Ceria, and Gérard Cornuéjols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Math. Program.*, 58(3):295–324, February 1993.

9 Manisha Bansal, Naveen Garg, and Neelima Gupta. A 5-approximation for capacitated facility location. In Leah Epstein and Paolo Ferragina, editors, *Algorithms – ESA 2012*, volume 7501 of *Lecture Notes in Computer Science*, pages 133–144. Springer Berlin Heidelberg, 2012.

10 Nikhil Bansal and Maxim Sviridenko. The Santa Claus problem. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, STOC'06, pages 31–40, New York, NY, USA, 2006. ACM.

11 Siu On Chan, James R. Lee, Prasad Raghavendra, and David Steurer. Approximate constraint satisfaction requires large lp relaxations. *CoRR*, abs/1309.0563, 2013.

12 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Integrality gaps for Sherali-Adams relaxations. In *Proceedings of the 41st annual ACM Symposium on Theory of Computing*, STOC'09, pages 283–292, New York, NY, USA, 2009. ACM.

13 Gérard Cornuéjols. Valid inequalities for mixed integer linear programs. *Math. Program.*, 112(1):3–44, 2008.

14 Marek Cygan, MohammadTaghi Hajiaghayi, and Samir Khuller. LP rounding for k-centers with non-uniform hard capacities. In *FOCS*, pages 273–282, 2012.

15 Wenceslas Fernandez de la Vega and Claire Kenyon-Mathieu. Linear programming relaxations of maxcut. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA'07, pages 53–61, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

16 Konstantinos Georgiou and Avner Magen. Limitations of the Sherali-Adams lift and project system: Compromising local and global arguments. Technical Report CSRG-587, University of Toronto, 2008.

17 S. Guha and S. Khuller. Greedy strikes back: improved facility location algorithms. *Journal of Algorithms*, 31:228–248, 1999.

18 D. S. Hochbaum. Heuristics for the fixed cost median problem. *Mathematical Programming*, 22:148–162, 1982.

19 Kamal Jain, Mohammad Mahdian, Evangelos Markakis, Amin Saberi, and Vijay V. Vazirani. Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. *J. ACM*, 50(6):795–824, November 2003.

20 Stavros G. Kolliopoulos and Yannis Moysoglou. Exponential lower bounds on the size of approximate formulations in the natural encoding for capacitated facility location. *CoRR*, abs/1312.1819, 2013.

21 Stavros G. Kolliopoulos and Yannis Moysoglou. Tight bounds on the Lovász-Schrijver rank for approximate capacitated facility location. *Manuscript*, 2013.

22 Jean B. Lasserre. An explicit exact SDP relaxation for nonlinear 0-1 programs. In *Proceedings of the 8th International IPCO Conference on Integer Programming and Combinatorial Optimization*, pages 293–303, London, UK, UK, 2001. Springer-Verlag.

23 Monique Laurent. A Comparison of the Sherali-Adams, Lovász-Schrijver, and Lasserre relaxations for 0–1 Programming. *Math. Oper. Res.*, 28(3):470–496, July 2003.

24 J. M. Y. Leung and T. L. Magnanti. Valid inequalities and facets of the capacitated plant location problem. *Math. Program.*, 44(1-3):271–291, 1989.

25 Retsef Levi, David B. Shmoys, and Chaitanya Swamy. LP-based approximation algorithms for capacitated facility location. *Math. Program.*, 131(1-2):365–379, 2012. Preliminary version in Proc. IPCO 2004.

26 Shi Li. A 1.488 approximation algorithm for the uncapacitated facility location problem. *Inf. Comput.*, 222:45–58, January 2013.

27 Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Proc. 45th STOC*, pages 901–910. ACM, 2013.

28 L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1:166–190, 1991.

29 Grant Schoenebeck, Luca Trevisan, and Madhur Tulsiani. Tight integrality gaps for Lovasz-Schrijver LP relaxations of Vertex Cover and Max Cut. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, STOC'07, pages 302–310, New York, NY, USA, 2007. ACM.

30 Ola Svensson. Santa claus schedules jobs on unrelated machines. *SIAM J. Comput.*, 41(5):1318–1341, 2012.

31 Z. Svitkina. Lower-bounded facility location. In *Proceedings of the 19th ACM-SIAM Symposium on Discrete Algorithms*, pages 1154–1163, 2008.

32 J. Vygen. Approximation algorithms for facility location problems (Lecture Notes). Report 05950-OR, Research Institute for Discrete Mathematics, University of Bonn, 2005. URL: `http://www.or.uni-bonn.de/~vygen/files/fl.pdf`.

33 David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, New York, NY, USA, 1st edition, 2011.

Lower Bounds on Expansions of Graph Powers

Tsz Chiu Kwok and Lap Chi Lau

The Chinese University of Hong Kong
Shatin, Hong Kong
{tckwok,chi}@cse.cuhk.edu.hk

Abstract

Given a lazy regular graph G, we prove that the expansion of G^t is at least $\Omega(\sqrt{t})$ times the expansion of G. This bound is tight and can be generalized to small set expansion. We show some applications of this result.

1998 ACM Subject Classification G.2.2 Graph Theory

Keywords and phrases Conductance, Expansion, Graph power, Random walk

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.313

1 Introduction

Let $G = (V, E, w)$ be an undirected weighted graph with $n = |V|$ vertices. The expansion of a set $S \subseteq V$ is defined as

$$\phi(S) := \frac{1}{|S|} \sum_{u \in S, v \notin S} w(u, v),$$

and the expansion of G is defined as

$$\phi(G) := \min_{S \subseteq V, |S| \leq n/2} \phi(S).$$

Graph expansion is a fundamental parameter with diverse applications in theoretical computer science [4].

A well-known operation to improve the graph expansion is by taking the t-th power of G, which has a natural correspondence to simulating the random walk on G for t steps. In our setting, we assume that G is 1-regular, that is, $\sum_{v \in V} w(u, v) = 1$ for every $u \in V$. We also assume that G is lazy, that is, $w(u, u) \geq \frac{1}{2}$ for every $u \in V$. Let A be the adjacency matrix of G with $A_{u,v} = w(u, v)$ for any $u, v \in V$, which corresponds to the transition matrix of the random walk on G. The t-th power of G, denoted by G^t, is defined as the undirected graph with adjacency matrix A^t, which corresponds to the transition matrix of the t-step random walk of G. Note that G^t is also 1-regular if G is.

The question we study is to prove lower bounds on $\phi(G^t)$ in terms of $\phi(G)$. Besides being a basic graph theoretical question, proving lower bounds on $\phi(G^t)$ has applications in hardness of approximation [3, 8]. Our main result is a tight lower bound on the expansion of the graph power of a lazy 1-regular graph.

1.1 Previous Work

There is a spectral method to show that $\phi(G^t)$ is larger than $\phi(G)$ for large enough t. This is based on the connection between the graph expansion and the second eigenvalue of the adjacency matrix. Let $1 = \alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_n \geq 0$ be the eigenvalues of the adjacency

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 313–324

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

matrix A_G of G, where $\alpha_1 = 1$ because G is 1-regular and $\alpha_n \geq 0$ because G is lazy. Let $L_G := I - A_G$ be the Laplacian matrix of G and $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n \leq 1$ be its eigenvalues. Note that $\lambda_i = 1 - \alpha_i$ for $1 \leq i \leq n$. Cheeger's inequality [1] states that

$$\frac{1}{2}\lambda_2 \leq \phi(G) \leq \sqrt{2\lambda_2}.$$

Note that the eigenvalues of A^t is $1 = \alpha_1^t \geq \alpha_2^t \geq \ldots \geq \alpha_n^t \geq 0$, and thus the i-th eigenvalue of the Laplacian matrix of G^t is $1 - \alpha_i^t = 1 - (1 - \lambda_i)^t$. Therefore, by Cheeger's inequality, we have

$$\phi(G^t) \geq \frac{1}{2}(1 - (1 - \lambda_2)^t) \geq \frac{1}{2}(1 - (1 - \frac{1}{2}t\lambda_2)) = \frac{1}{4}t\lambda_2 \geq \frac{1}{8}t \cdot \phi(G)^2 = \Omega(t \cdot \phi(G)^2),$$

where the second inequality follows from Fact 2.1 when $t\lambda_2 < 1/2$.

Recently, the spectral method was extended to prove lower bounds on the small set expansion of a graph. Given $0 < \delta < 1/2$, the small set expansion of G is defined as

$$\phi_\delta(G) := \min_{S \subseteq V, |S| \leq \delta n} \phi(S).$$

Raghavendra and Schramm [8] proved an analog of the above bound for small set expansion:

$$\phi_{\Omega(\delta)}(G^t) = \Omega(t \cdot \phi_\delta(G)^2),$$

when G is a lazy 1-regular graph and $t = O(1/\phi_\delta(G)^2)$. The proof is based on the techniques developed in [2] relating higher eigenvalues to small set expansion. They used this lower bound to amplify the hardness of the small set expansion problem; see Section 3.3 for more discussions.

1.2 Our Results

Our main result is a tight lower bound on $\phi(G^t)$.

▶ **Theorem 1.** *Let G be an undirected 1-regular lazy graph. For any non-negative integer t, we have*

$$\phi(G^t) \geq \frac{1}{20}(1 - (1 - \phi(G))^{\sqrt{t}}) = \Omega(\min(\sqrt{t} \cdot \phi(G), 1)).$$

This is a quadratic improvement of the previous bound. This bound is tight up to a constant factor for all t as we will show examples (e.g. cycles) in Section 2.6.

Observe that the above spectral method only showed that $\phi(G^t) > \phi(G)$ when $t = \Omega(1/\phi(G))$ but did not show that $\phi(G^t) > \phi(G)$ for small t. Theorem 1 implies that $\phi(G^t) > \phi(G)$ for some small constant t. Actually, we can show that $\phi(G^3) > \phi(G)$ when $\phi(G) < 1/2$ by a more careful calculation.

▶ **Theorem 2.** *Let G be an undirected 1-regular lazy graph with even n. We have*

$$\phi(G^3) \geq \frac{3}{2}\phi(G) - 2\phi(G)^3.$$

Theorem 1 can be extended easily to small set expansion.

▶ **Theorem 3.** *Let G be an undirected 1-regular lazy graph. For any non-negative integer t, we have*

$$\phi_{\delta/2}(G^t) \geq \frac{1}{20}(1 - (1 - 2\phi_\delta(G))^{\sqrt{t}}) = \Omega(\min(\sqrt{t} \cdot \phi_\delta(G), 1)).$$

We show some applications of our results in Section 3, including the gap amplification result in [8] for small set expansion and some reductions for proving Cheeger-type inequalities [1, 6].

1.3 Techniques

Instead of using the spectral method, we use the Lovász-Simonovits curve [7] which was designed to analyze the mixing time of random walk using graph expansion. As it turns out, this more combinatorial approach has the advantage of directly reason about graph expansion without having the quadratic loss in the spectral method.

Given an initial probability distribution p on the vertex set, let $C^{(t)}(x)$ be the sum of the probability of the x largest vertices after t steps of random walk on G. First, we observe in Lemma 6 that

$$\phi_{G^t}(S) \geq 1 - C^{(t)}(|S|)$$

when the initial distribution is $\chi_S/|S|$ where χ_S is the characteristic vector of S. Hence, to lower bound $\phi_{G^t}(S)$, we can instead upper bound $C^{(t)}(|S|)$. Imprecisely, with the method developed by Lovász and Simonovits (see Section 2.2), we can essentially argue that for all S with $|S| \leq n/2$,

$$\begin{aligned} C^{(t)}(|S|) &\lesssim \frac{1}{2^t}\sum_{i=0}^{t}\binom{t}{i} C^{(0)}\big((1-\phi(G))^i(1+\phi(G))^{t-i}|S|\big) \\ &= \frac{1}{2^t}\sum_{i=0}^{t}\binom{t}{i} \min\{(1-\phi(G))^i(1+\phi(G))^{t-i}, 1\}, \end{aligned}$$

where the equality holds because $C^{(0)}(x) = \min\{x/|S|, 1\}$ as the initial distribution is $\chi_S/|S|$. Since there is at least a $1/10$ fraction of terms in the summation with $i \geq t/2 + \sqrt{t}$, we have

$$C^{(t)}(|S|) \lesssim \frac{1}{10}(1-\phi(G))^{\sqrt{t}} + \frac{9}{10} \leq \frac{1}{10}(1 - \frac{1}{2}\sqrt{t}\cdot\phi(G)) + \frac{9}{10},$$

where the last inequality is by Fact 2.1 when $\sqrt{t}\cdot\phi(G) \leq 1/2$. Therefore, for all S with $|S| \leq n/2$, we have

$$\phi_{G^t}(S) \geq \frac{1}{20}\sqrt{t}\cdot\phi(G), \text{ and therefore } \phi(G^t) = \Omega(\sqrt{t}\cdot\phi(G)).$$

We need to be careful to make the arguments in $\lesssim$ precise and this is some technicality of the proof, but the main ideas are pretty accurately summarized in this section.

2 Expansion of Graph Power

2.1 Preliminaries

When G is clear from the context, we use $\phi = \phi(G)$ to denote the conductance of G.

Let χ_S be a column vector such that $\chi_S(u) = 1$ if $u \in S$ and $\chi_S(u) = 0$ otherwise. The expansion of S can be expressed as

$$\phi(S) = \frac{\chi_S^T(I-A)\chi_S}{|S|}.$$

The following fact is used frequently in the proof.

▶ **Fact 2.1.** *For any $z \in [0,1]$, we have*

$$(1-z)^t \geq 1 - zt, \quad \text{or} \quad 1-(1-z)^t \leq zt.$$

For any $zt \in [0, 1/2]$, we have

$$(1-z)^t \leq \exp(-zt) \leq 1 - \frac{1}{2}zt, \quad \text{or} \quad 1-(1-z)^t \geq \frac{1}{2}zt.$$

2.2 Lovász-Simonovits Curve

Lovász and Simonovits [7] introduced a curve which is useful in bounding mixing time using graph expansion. Given a probabilistic vector $p : V \to \mathbb{R}_{\geq 0}$, the curve is defined as

$$C(p, x) = \max_{\delta_1 + \cdots + \delta_n = x, 0 \leq \delta_i \leq 1} \sum_{i=1}^{n} \delta_i p_i,$$

for $x \in [0, n]$. When x is an integer, $C(p, x)$ is simply the sum of the largest x values in the vector p, and it is linear between two integral values. Clearly $C(p, x)$ is concave. We use $C^{(t)}(x)$ to denote $C(A^t p, x)$ when p is clear from the context. We use $\overline{x}$ to denote $\min(x, n - x)$ for $x \in [0, n]$. This notation is frequently used and should be interpreted as the distance to the boundary. The following lemma shows that the curves "drop" faster when the expansion of G is larger.

▶ **Lemma 4** ([7]). *If G is a lazy 1-regular graph, then for any integer $t \geq 0$ and any integer $x \in [0, n]$, we have*

$$C^{(t+1)}(x) \leq \frac{1}{2}\left(C^{(t)}(x - 2\phi\overline{x}) + C^{(t)}(x + 2\phi\overline{x})\right).$$

We remark that Lemma 4 only give bounds on integral values[1]. In our proof, however, we require bounds for all $x \in [0, n]$. The following lemma provides a slightly weaker bound that also holds for fractional x when the graph is lazy 1-regular.

▶ **Lemma 5.** *If G is a lazy 1-regular graph, then for any integer $t \geq 0$ and $x \in [0, n]$, we have*

$$C^{(t+1)}(x) \leq \frac{1}{2}\left(C^{(t)}(x - \phi\overline{x}) + C^{(t)}(x + \phi\overline{x})\right).$$

Proof. Since $C^{(t)}$ is concave, we have

$$C^{(t)}(x - \beta\overline{x}) + C^{(t)}(x + \beta\overline{x}) \leq C^{(t)}(x - \gamma\overline{x}) + C^{(t)}(x + \gamma\overline{x}) \text{ for } \beta > \gamma. \tag{1}$$

We will prove that

$$C^{(t+1)}(x) \leq \frac{1}{2}\left(C^{(t)}(x - 2\phi'\overline{x}) + C^{(t)}(x + 2\phi'\overline{x})\right) \tag{2}$$

where $\phi' = \frac{n-1}{n}\phi$, and this would imply the lemma by (1) since $\phi' \geq \frac{1}{2}\phi$.

Note that for any integral $x \in [0, n-1]$ and any $\alpha \in [0, 1]$,

$$\begin{aligned}
C^{(t+1)}(x + \alpha) &= (1 - \alpha)C^{(t+1)}(x) + \alpha C^{(t+1)}(x + 1) \\
&\leq (1 - \alpha)\left(C^{(t)}(x - 2\phi\overline{x}) + C^{(t)}(x + 2\phi\overline{x})\right) \\
&\qquad + \alpha\left(C^{(t)}(x + 1 - 2\phi\overline{(x+1)}) + C^{(t)}(x + 1 + 2\phi\overline{(x+1)})\right) \\
&= \left((1 - \alpha)C^{(t)}(x - 2\phi\overline{x}) + \alpha C^{(t)}(x + 1 - 2\phi\overline{(x+1)})\right) \\
&\qquad + \left((1 - \alpha)C^{(t)}(x + 2\phi\overline{x}) + \alpha C^{(t)}(x + 1 + 2\phi\overline{(x+1)})\right) \\
&\leq C^{(t)}(x + \alpha - 2\phi((1 - \alpha)\overline{x} + \alpha\overline{(x+1)})) \\
&\qquad + C^{(t)}(x + \alpha + 2\phi((1 - \alpha)\overline{x} + \alpha\overline{(x+1)})),
\end{aligned}$$

[1] It was claimed in [7] that the lemma holds for any $x \in [0, n]$, but later it was pointed out in [10] that the lemma only holds for integral x when the graph is lazy 1-regular.

where the first inequality follows from Lemma 4, and last inequality holds because $C^{(t)}$ is concave. If $(1-\alpha)\overline{x}+\alpha\overline{(x+1)} = \overline{(x+\alpha)}$, then Lemma 4 holds and the lemma follows by (1).

Note that the only case where $(1-\alpha)\overline{x}+\alpha\overline{(x+1)} \neq \overline{(x+\alpha)}$ is when n is odd and $x=(n-1)/2$. At that time, $\overline{x}=\overline{(x+1)}=x$ and thus $(1-\alpha)\overline{x}+\alpha\overline{(x+1)}=x$. Therefore, when n is odd and $x=(n-1)/2$, we have

$$\begin{aligned}
&C^{(t+1)}(x+\alpha)\\
&\leq \frac{1}{2}\left(C^{(t)}(x+\alpha-2\phi x)+C^{(t)}(x+\alpha+2\phi x)\right)\\
&\leq \frac{1}{2}\left(C^{(t)}\left(x+\alpha-2(\frac{n-1}{n})\cdot\phi\cdot\overline{(x+\alpha)}\right)+C^{(t)}\left(x+\alpha+2(\frac{n-1}{n})\cdot\phi\cdot\overline{(x+\alpha)}\right)\right)\\
&= \frac{1}{2}\left(C^{(t)}\left(x+\alpha-2\phi'\cdot\overline{(x+\alpha)}\right)+C^{(t)}\left(x+\alpha+2\phi'\cdot\overline{(x+\alpha)}\right)\right),
\end{aligned}$$

where the later inequality holds because $C^{(t)}$ is concave and $\overline{x+\alpha}\leq x+\frac{1}{2}=n/2$. ◀

2.3 Proof of Theorem 1

As mentioned in the proof outline in Section 1.3, we first show that we can prove a lower bound on $\phi(G^t)$ by proving an upper bound on $C^{(t)}(|S|)$ for the initial distribution $\chi_S/|S|$.

▶ **Lemma 6.** *Suppose that for any set $S\subseteq V$ with $|S|\leq n/2$, we have $C^{(t)}(|S|)\leq 1-\alpha$ for the initial distribution $p=\chi_S/|S|$, then we can conclude that $\phi(G^t)\geq\alpha$.*

Proof. Let S be the set attaining minimum expansion in G^t, that is, $|S|\leq n/2$ and $\phi_{G^t}(S)=\phi(G^t)$. For the initial distribution $p=\chi_S/|S|$,

$$C^{(t)}(|S|)=C(A^tp,|S|)\geq\chi_S^TA^tp=\frac{\chi_S^TA^t\chi_S}{|S|}=1-\frac{\chi_S^T(I-A^t)\chi_S}{|S|}=1-\phi_{G^t}(S).$$

Therefore, we have $\phi(G^t)=\phi_{G^t}(S)\geq 1-C^{(t)}(|S|)\geq\alpha$. ◀

With Lemma 6, it remains to upper bound $C^{(t)}(|S|)$ for the initial distribution $\chi_S/|S|$ for any S with $|S|\leq n/2$. It turns out that there is a good upper bound independent of $|S|$.

▶ **Lemma 7.** *For any S with $|S|\leq n/2$, for the initial distribution $p=\chi_S/|S|$, for any non-negative integer t, we have*

$$C^{(t)}(|S|)\leq 1-\frac{1}{20}(1-(1-\phi)^{\sqrt{t}}).$$

Proof. For technical reasons, we consider $D^{(t)}(x)=C^{(t)}(x)-x/n$ instead to make the argument more symmetric. See Figure 1 for the definition of $D^{(0)}$. Note that Lemma 5 still holds for $D^{(t)}$ since x/n is linear. So, we have

$$D^{(t+1)}(x)\leq\frac{1}{2}\left(D^{(t)}(x-\phi\overline{x})+D^{(t)}(x+\phi\overline{x})\right).$$

By applying this equation repeatedly, we have

$$D^{(t)}(x)\leq\frac{1}{2^t}\sum_{T\in\{-1,1\}^t}D^{(0)}(f_T(x)), \tag{3}$$

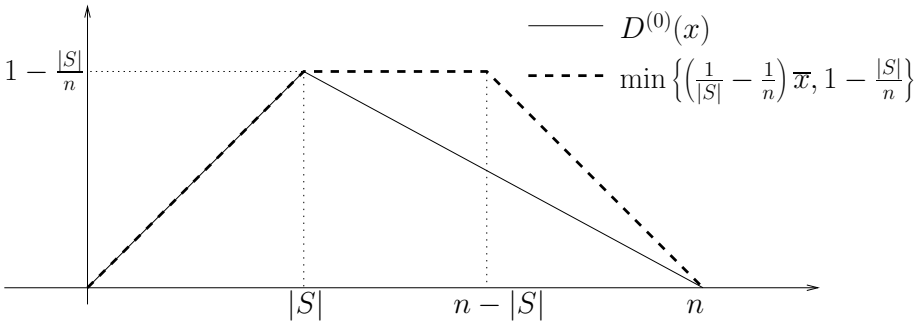


Figure 1 The solid line is the curve $D^0(x)$ and the dotted line is the upper bound on $D^{(0)}(x)$ that is stated in (5).

where T is a sequence of t ± 1-bits and f_T is defined recursively as follows. In the base case, when the sequence is empty, we define $f_{()}(x) = x$ for any $x \in [0, n]$. For any partial sequence T', we define

$$f_{(T',+1)}(x) = \begin{cases} f_{T'}(x) - \phi \cdot \overline{f_{T'}(x)} & \text{if } f_{T'}(x) \le n/2 \\ f_{T'}(x) + \phi \cdot \overline{f_{T'}(x)} & \text{if } f_{T'}(x) > n/2\,, \end{cases}$$

and

$$f_{(T',-1)}(x) = \begin{cases} f_{T'}(x) + \phi \cdot \overline{f_{T'}(x)} & \text{if } f_{T'}(x) \le n/2 \\ f_{T'}(x) - \phi \cdot \overline{f_{T'}(x)} & \text{if } f_{T'}(x) > n/2\,. \end{cases}$$

We can view $+1$ as moving in the direction towards boundary and -1 as moving in the direction towards center. Recall that $\overline{x} = \min\{x, n-x\}$ can be viewed as the distance to the boundary. In the following, we focus on the distance to the boundary of a point rather than its actual location. It follows from the definition that for any $x \in [0, n]$, we have

$$\overline{f_{+1}(x)} = \overline{x} - \phi\overline{x} = (1-\phi) \cdot \overline{x},$$

and

$$\overline{f_{-1}(x)} \le \overline{x} + \phi\overline{x} = (1+\phi)\overline{x} \le (1-\phi)^{-1} \cdot \overline{x}.$$

Therefore, $\overline{f_{T_i}(x)} \le (1-\phi)^{T_i} \cdot \overline{x}$ where T_i is the i-th bit in the sequence T, and hence

$$\overline{f_T(x)} = \overline{f_{T_t} \circ f_{T_{t-1}} \circ \cdots \circ f_{T_1}(x)} \le (1-\phi)^{T_t} \cdot \overline{f_{T_{t-1}} \circ \cdots \circ f_{T_1}(x)} \le \cdots \le (1-\phi)^{\sum_{i=1}^{t} T_i} \cdot \overline{x}.$$

We call a sequence T good if $\sum_{i=1}^{t} T_i \ge \sqrt{t}$, otherwise we call it bad.

For a good T, we have $\overline{f_T(x)} \le (1-\phi)^{\sqrt{t}} \cdot \overline{x}$, and thus

$$\overline{f_T(|S|)} \le (1-\phi)^{\sqrt{t}} \cdot |S| \quad \text{for } |S| \le n/2 \text{ and } T \text{ good.} \tag{4}$$

As the initial distribution is $\chi_S/|S|$, for $t = 0$, we have

$$D^{(0)}(x) \le \min\left\{\left(\frac{1}{|S|} - \frac{1}{n}\right)\overline{x}, 1 - \frac{|S|}{n}\right\}. \tag{5}$$

See Figure 1 for an illustration of the inequality. The advantage of using $D^{(t)}$ instead of $C^{(t)}$ is that we could bound $D^{(0)}(x)$ using $\overline{x}$ as shown in the above inequality.

Finally, we know that at least a $1/10$ fraction of T are good. So, for S with $|S| \le n/2$,

$$\begin{aligned}
D^{(t)}(|S|) &\le \frac{1}{2^t} \sum_{T\in\{-1,1\}^t} D^{(0)}(f_T(|S|)) && \text{(by (3))}\\
&= \frac{1}{2^t} \sum_{T:\text{good}} D^{(0)}(f_T(|S|)) + \frac{1}{2^t} \sum_{T:\text{bad}} D^{(0)}(f_T(|S|))\\
&\le \frac{1}{2^t} \sum_{T:\text{good}} \left(\frac{1}{|S|} - \frac{1}{n}\right) \overline{f_T(|S|)} + \frac{1}{2^t} \sum_{T:\text{bad}} \left(1 - \frac{|S|}{n}\right) && \text{(by (5))}\\
&\le \frac{1}{2^t} \sum_{T:\text{good}} \left(\frac{1}{|S|} - \frac{1}{n}\right) (1-\phi)^{\sqrt{t}}|S| + \frac{1}{2^t} \sum_{T:\text{bad}} \left(1 - \frac{|S|}{n}\right) && \text{(by (4))}\\
&\le \frac{1}{10}\left(\left(\frac{1}{|S|} - \frac{1}{n}\right) (1-\phi)^{\sqrt{t}}|S|\right) + \frac{9}{10}\left(1 - \frac{|S|}{n}\right)\\
&= \left(1 - \frac{|S|}{n}\right) - \frac{1}{10}\left(1 - \frac{|S|}{n} - \left(\frac{1}{|S|} - \frac{1}{n}\right)(1-\phi)^{\sqrt{t}}|S|\right)\\
&= \left(1 - \frac{|S|}{n}\right) - \frac{1}{10}\left(1 - \frac{|S|}{n}\right)\left(1 - (1-\phi)^{\sqrt{t}}\right)\\
&\le \left(1 - \frac{|S|}{n}\right) - \frac{1}{20}(1 - (1-\phi)^{\sqrt{t}}).
\end{aligned}$$

Therefore,

$$C^{(t)}(|S|) = D^{(t)}(|S|) + |S|/n \le 1 - \frac{1}{20}(1 - (1-\phi)^{\sqrt{t}}).$$

◀

Combining Lemma 6 and Lemma 7, we have

$$\phi(G^t) \ge \frac{1}{20}(1 - (1-\phi)^{\sqrt{t}}) \ge \frac{1}{40}\sqrt{t}\cdot\phi,$$

where the last inequality is by Fact 2.1 for $\sqrt{t}\cdot\phi \le 1/2$. This completes the proof of Theorem 1.

2.4 Proof of Theorem 2

Theorem 1 showed that $\phi(G^t) > \phi(G)$ for a small constant t. To prove that this is true even for $t = 3$, we need to do a more careful calculation. We use the bound

$$C^{(t+1)}(x) \le \frac{1}{2}\left(C^{(t)}(x - 2\phi'\overline{x}) + C^{(t)}(x + 2\phi'\overline{x})\right)$$

for $\phi' = \frac{n-1}{n}\phi$ as was shown in (2) in the proof of Lemma 5. When $t = 3$, we have

$$\begin{aligned}
C^{(3)}(|S|) &\le \frac{1}{8}C^{(0)}((1-2\phi')^3|S|) + \frac{3}{8}C^{(0)}((1-2\phi')^2(1+2\phi')|S|) + \frac{4}{8}\\
&= \frac{1}{8}(1-2\phi')^3 + \frac{3}{8}(1-2\phi')^2(1+2\phi') + \frac{4}{8}\\
&= 1 - \frac{3}{2}\phi' + 2\phi'^3.
\end{aligned}$$

Thus we conclude $\phi(G^3) \ge \frac{3}{2}\phi' - 2\phi'^3$. Therefore, for a large graph with small conductance, taking cube increases the conductance by a factor of almost $\frac{3}{2}$. When n is even, we can replace ϕ' by ϕ as was shown in the proof of Lemma 5, and this proves Theorem 2.

2.5 Proof of Theorem 3

Our result can be easily extended to the case of small set expansion with a little loss in size. More precisely, suppose G is an undirected 1-regular lazy graph such that all sets of size at most δn have conductance ϕ_δ, where $\delta \leq 1/2$. In this setting, the following lemma holds in place of Lemma 4.

▶ **Lemma 8.** *If G is a lazy 1-regular graph, then for any integer $t \geq 0$ and any $x \in [0, \delta n]$,*

$$C^{(t+1)}(x) \leq \frac{1}{2}\left(C^{(t)}(x - 2\phi_\delta \cdot \overline{x}) + C^{(t)}(x + 2\phi_\delta \cdot \overline{x})\right),$$

where $\overline{x} = \min(x, \delta n - x)$ here.

We remark that we do not need to fix the non-integral problem as in Lemma 5 because we only consider $x \leq \delta n \leq n/2$ (see the proof of Lemma 5).

Lemma 6 can be restated as follows with the same proof.

▶ **Lemma 9.** *Suppose that for any set $S \subseteq V$ with $|S| \leq \delta n/2$ with the initial distribution $p = \chi_S/|S|$, we have $C^{(t)}(|S|) \leq 1 - \alpha$, then we can conclude that $\phi_{\delta/2}(G^t) \geq \alpha$.*

Finally, in Lemma 7, we consider $D^{(t)}(x) = C^{(t)}(x) - \frac{x}{\delta n}$ instead, and we use the new $\overline{x}$ in the analysis. Observe that $f_T(x)$ can never leave the range $[0, \delta n]$ when x starts in the range. Therefore the same analysis applies and we have the following lemma.

▶ **Lemma 10.** *For any S with $|S| \leq \delta n/2$, for the initial distribution $p = \chi_S/|S|$, for any non-negative integer t, we have*

$$C^{(t)}(|S|) \leq 1 - \frac{1}{20}(1 - (1 - 2\phi_\delta)^{\sqrt{t}}).$$

Theorem 3 follows by combining Lemma 9 and Lemma 10.

2.6 Tight Examples

We show that the dependence on t in Theorem 1 is tight up to a constant factor. The tight example we use is a lazy cycle. Intuitively, after t steps of random walk on a lazy cycle, the final position with high probability only differs from the initial position by $O(\sqrt{t})$, and therefore the expansion should be bounded by $O(\sqrt{t})$ times the original expansion. It turns out that we can easily justify this intuition through Cheeger's inequality.

▶ **Proposition 2.2.** *Let C_n be the lazy cycle. Then we have $\phi(C_n^t) = O(\sqrt{t} \cdot \phi(C_n))$.*

Proof. As in Section 1.1, we have

$$\lambda_2(C_n^t) = 1 - (1 - \lambda_2(C_n))^t \leq t\lambda_2(C_n) = O(t \cdot \phi(C_n)^2),$$

where the inequality is by Fact 2.1 and the last equality is by the spectrum of the cycle. By Cheeger's inequality, $\phi(C_n^t) = O(\sqrt{\lambda_2(C_n^t)})$, and thus $\phi(C_n^t) = O(\sqrt{t} \cdot \phi(C_n))$. ◀

We remark that tight examples of Theorem 1 must have "high threshold rank". By the improved Cheeger's inequality in [6], we have $\phi(G) = O(k\lambda_2/\sqrt{\lambda_k})$ for any k. Therefore, by the same calculation as in Section 1.1, we have that for any k,

$$\phi(G^t) \geq \frac{1}{4}t\lambda_2 = \Omega(\frac{t \cdot \phi(G) \cdot \sqrt{\lambda_k}}{k}),$$

and therefore a graph G with $\lambda_k(G)$ small for a small k could not be a tight example for Theorem 1.

2.7 Irregular Graphs

Theorem 1 showed that $\phi(G^t) = \Omega(\sqrt{t} \cdot \phi(G))$ for a regular graph. There are different ways to generalize the statement to irregular graphs. In the following, we show that the generalization is not true if we replace expansion by conductance, and we show that the generalization is true if we replace expansion by the escape probability of a t-step random walk.

The conductance of a set $\phi(S)$ is defined as

$$\frac{\sum_{v \in S, u \notin S} w(u,v)}{\mathrm{vol}(S)},$$

where $\mathrm{vol}(S) := \sum_{v \in S} \deg(v)$ and the conductance of a graph $\phi(G)$ is defined as

$$\min_{S \subseteq V, \mathrm{vol}(S) \leq \mathrm{vol}(V)/2} \phi(S).$$

Consider the graph G consisting of a regular complete graph with self loops $(2I + \frac{1}{n}K_n)$ and an extra vertex u. The extra vertex only connects to a single vertex v in the complete graph with edge weight 1 and it has a self loop of weight m. We assume the complete graph is so large that $n > 2m^4$. Then $\phi(G) = \phi(\{u\}) = 1/m + o(1/m)$. Consider G^3. Since $\deg_{G^3}(u) = m^3 + o(m^3) < n/2$, the set achieving minimum conductance is still $\{u\}$. In G^3, the total weight of edges between u and the complete graph is $m^2 + o(m^2)$. Therefore $\phi(G^3) = 1/m + o(1/m)$. Note that the same argument applies for any G^t if we set n to be large enough. Therefore, no matter how small $\phi(G)$ is or how large t is, we cannot argue that $\phi(G^t) > (1+\epsilon)\phi(G)$ for a positive constant ϵ when we replace expansion by conductance in irregular graphs.

On the other hand, our results can be extend to another natural generalization of expansion. Consider the definition

$$\varphi(G^t) = \min_{S \subseteq V, |S| \leq n/2} \varphi_{G^t}(S) = \min_{S \subseteq V, |S| \leq n/2} (1 - \frac{\chi_S^T (D^{-1}A_G)^t \chi_S}{|S|}),$$

where $\varphi_{G^t}(S)$ is the probability that a t-step random walk starting from a random vertex in S escapes S. With this definition and assuming that the graph does not contain a vertex of degree more than half of the total degrees, we can show that Lemma 5 still holds, with a extended definition for $C^{(t)}$. Therefore, $\varphi(G^t) = \Omega(\min\{\sqrt{t} \cdot \varphi(G), 1\})$ follows.

3 Applications

In this section, we discuss some consequences of our main theorem. We show that proving the general cases of Cheeger's inequalities can be reduced to proving the special cases where the eigenvalues are constants. Similar arguments can be used to deduce the recent result on gap amplification of small set expansion in [8].

3.1 Cheeger's Inequalities

Let G be an undirected 1-regular lazy graph. The following result shows that if one could prove Cheeger's inequality when λ_2 is a constant, then one could prove Cheeger's inequality for all λ_2. One consequence is that if one could prove that say $\phi(G) = O((\lambda_2)^{1/100})$ (so that Cheeger's inequality is true when λ_2 is a constant), then it actually implies that $\phi(G) = O(\sqrt{\lambda_2})$.

▶ **Corollary 11.** *Suppose one could prove that $\lambda_2(H) \geq C$ for some constant $C \leq 1/2$ whenever $\phi(H) \geq 1/40$, then it implies that $\phi(G) \leq \sqrt{\lambda_2(G)/C}$ for any G and any $\lambda_2(G)$.*

Proof. Given G, we assume that $\lambda_2(G) \leq \phi(G)^2/2$, as otherwise the statement is trivial. Consider $H = G^{1/\phi(G)^2}$. By Theorem 1, we have

$$\phi(H) \geq \frac{1}{20}(1 - (1 - \phi(G))^{\sqrt{1/\phi(G)^2}}) \geq \frac{1}{40}.$$

Therefore, if we could prove that $\lambda_2(H) \geq C$, then we could conclude that

$$C \leq \lambda_2(H) = 1 - (1 - \lambda_2(G))^{1/\phi(G)^2} \leq \frac{\lambda_2(G)}{\phi(G)^2},$$

and the corollary follows. ◀

3.2 Improved Cheeger's Inequality

It was proved in [6] that $\phi(G) = O(k\lambda_2/\sqrt{\lambda_k})$ for any k. Using similar arguments as above, the following result shows that if one could prove this improved Cheeger's inequality when λ_3 is a constant, then one could prove it for all λ_3. For instance, if one could prove that say $\phi(G) = O(\lambda_2/\lambda_3^{100})$, then it actually implies that $\phi(G) = O(\lambda_2/\sqrt{\lambda_3})$.

▶ **Corollary 12.** *Suppose one could prove that $\phi(H) \leq C\lambda_2(H)$ for some $C \geq 1/10$ whenever $\lambda_3(H) \geq 1/2$, then it implies that $\phi(G) \leq 40C\lambda_2(G)/\sqrt{\lambda_3(G)}$ for any G and any $\lambda_3(G)$.*

Proof. We assume that $\phi \leq \sqrt{\lambda_3}/2$, as otherwise, by Cheeger's inequality, $2\lambda_2(G) \geq \phi(G)^2 \geq \frac{1}{2}\phi(G)\sqrt{\lambda_3}$ and the statement is true. Consider $H = G^{1/\lambda_3(G)}$. Then

$$\lambda_3(H) = 1 - (1 - \lambda_3(G))^{1/\lambda_3} \geq 1 - e^{-1} \geq 1/2.$$

Therefore, if one could prove that $\phi(H) \leq C\lambda_2(H)$, then

$$C\lambda_2(H) \geq \phi(H) \geq \frac{1}{20}(1 - (1 - \phi(G))^{\sqrt{1/\lambda_3(G)}}) \geq \frac{\phi(G)}{40\sqrt{\lambda_3(G)}},$$

where the second inequality is by Theorem 1 and the last inequality is by Fact 2.1. On the other hand,

$$\lambda_2(H) = 1 - (1 - \lambda_2(G))^{1/\lambda_3(G)} \leq \frac{\lambda_2(G)}{\lambda_3(G)},$$

and the corollary follows by combining the two inequalities. ◀

3.3 Gap Amplification for Small Set Expansion

Consider the small set expansion problem $\mathrm{SSE}_{\delta,\delta'}(c, s)$: Given a graph G, distinguish whether $\phi_\delta(G) \leq c$ or $\phi_{\delta'}(G) \geq s$. The small set expansion conjecture [9] states that for any $\epsilon > 0$, there exists $\delta > 0$ such that $\mathrm{SSE}_{\delta,\delta}(\epsilon, 1 - \epsilon)$ is NP-hard.

Let f be a function such that $f(x) = \omega(\sqrt{x})$. Raghavendra and Schramm [8] showed that if for all $\epsilon > 0$ there exists $\delta > 0$ such that $\mathrm{SSE}_{\delta,\delta}(\epsilon, f(\epsilon))$ is NP-hard, then for all $\epsilon > 0$ there exists $\delta > 0$ such that $\mathrm{SSE}_{\delta,\delta/8}(\epsilon, 1/2)$ is NP-hard.

We would show that our techniques can be easily applied to get similar result.

▶ **Theorem 13.** *If for all $\epsilon > 0$ there exists $\delta > 0$ such that $\mathrm{SSE}_{\delta,\delta}(\epsilon, f(\epsilon))$ is NP-hard, then for all $\epsilon > 0$ there exists $\delta > 0$ such that $\mathrm{SSE}_{\delta,\delta/2}(\epsilon, \Omega(1))$ is NP-hard.*

Proof. Given an instance G that we would like to distinguish whether $\phi_\delta(G) \leq \epsilon$ or $\phi_\delta(G) \geq f(\epsilon)$, we consider the graph $H = G^{O(1/f(\epsilon)^2)}$. In the case when $\phi_\delta(G) \geq f(\epsilon)$, by Theorem 3, we have

$$\phi_{\delta/2}(H) = \Omega(\sqrt{1/f(\epsilon)^2} \cdot f(\epsilon)) = \Omega(1).$$

In the case when $\phi_\delta(G) \leq \epsilon$, we have

$$\phi_\delta(H) \leq (1/f(\epsilon)^2) \cdot \epsilon = o_\epsilon(1) \leq \epsilon',$$

where the equality holds because $f(\epsilon) = \omega(\sqrt{\epsilon})$ and the first inequality holds because

$$\phi_{G^t}(S) = 1 - \frac{\chi_S^T A^t \chi_S}{|S|} \leq t \cdot \phi_G(S),$$

where the inequality is proven in [10] by a simple induction. Therefore, if $\mathrm{SSE}_{\delta,\delta}(\epsilon, f(\epsilon))$ is NP-hard, then $\mathrm{SSE}_{\delta,\delta/2}(\epsilon', \Omega(1))$ is NP-hard. ◀

Finally, we remark that it is easier to bound $\phi_\delta(G^t)$ for large t using Lovász-Simonovits curve. Using the techniques in [5], we have the following bound for $C^{(t)}$ when the initial probability vector is $\chi_S/|S|$:

$$C^{(t)}(x) \leq \frac{x}{\delta n} + \sqrt{\frac{x}{|S|}}(1 - \frac{\phi^2}{2})^t.$$

Therefore,

$$\phi_{G^t}(S) \geq 1 - C^{(t)}(|S|) \geq 1 - \frac{|S|}{\delta n} - (1 - \frac{\phi^2}{2})^t,$$

where the first inequality follows from Lemma 6. Set $t = 100/\phi^2$, then for $|S| \leq \delta n/4$, we have $\phi_{G^t}(S) \geq \frac{3}{4} - \exp(-50)$. Therefore, if $\mathrm{SSE}_{\delta,\delta}(\epsilon, f(\epsilon))$ is NP-hard, then $\mathrm{SSE}_{\delta,\delta/4}(\epsilon', 1/2)$ is NP-hard.

Acknowledgement. This research is supported by HK RGC grant 2150701.

References

1. N. Alon, V. Milman. *Isoperimetric inequalities for graphs, and superconcentrators.* Journal of Combinatorial Theory, Series B, 38(1), 73–88, 1985.
2. S. Arora, B. Barak, D. Steurer. *Subexponential algorithms for unique games and related problems.* In Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS), 563–572, 2010.
3. I. Dinur. *The PCP theorem by gap amplification.* Journal of the ACM 54(3), 12, 2007.
4. S. Hoory, N. Linial, A. Wigderson. *Expander graphs and their applications.* Bulletin of the American Mathematical Society 43(4), 439–561, 2006.
5. T. C. Kwok, L. C. Lau. *Finding small sparse cuts by random walk.* In Proceedings of the 16th International Workshop on Randomization and Computation (RANDOM), 615–626, 2012.
6. T. C. Kwok, L. C. Lau, Y. T. Lee, S. Oveis Gharan, L. Trevisan. *Improved Cheeger's inequality: analysis of spectral partitioning algorithms through higher order spectral gap.* In Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC), 11–20, 2013.

7 L. Lovász, M. Simonovits. *The mixing rate of Markov chains, an isoperimetric inequality, and computing the volume.* In Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science (FOCS), 346–354, 1990.

8 P. Raghavendra, T. Schramm. *Gap amplification for small-set expansion via random walk.* In Proceedings of the 17th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX), 2014.

9 P. Raghavendra, D. Steurer. *Graph expansion and the unique games conjecture.* In Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC), 755–764, 2010.

10 D. A. Spielman, S.-H. Teng. *A local clustering algorithm for massive graphs and its applications to nearly-linear time graph partitioning.* SIAM Journal on Computing 42(1), 1–26, 2013.

An Improved Algorithm for the Hard Uniform Capacitated k-median Problem

Shanfei Li

Delft Institute of Applied Mathematics, TU Delft, The Netherlands
shanfei.li@tudelft.nl

Abstract

In the k-median problem, given a set of locations, the goal is to select a subset of at most k centers so as to minimize the total cost of connecting each location to its nearest center. We study the uniform hard capacitated version of the k-median problem, in which each selected center can only serve a limited number of locations.

Inspired by the algorithm of Charikar, Guha, Tardos and Shmoys, we give a $(6 + 10\alpha)$-approximation algorithm for this problem with increasing the capacities by a factor of $2+\frac{2}{\alpha}, \alpha \geq 4$, which improves the previous best $(32l^2 + 28l + 7)$-approximation algorithm proposed by Byrka, Fleszar, Rybicki and Spoerhase violating the capacities by factor $2 + \frac{3}{l-1}, l \in \{2, 3, 4, \dots\}$.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Approximation algorithm, k-median problem, LP-rounding, Hard capacities

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.325

1 Introduction

In the capacitated k-median problem (CKM), we are given a set N of locations (where a center can potentially be opened). Each location $j \in N$ has a capacity M (uniform capacities), and a demand d_j that must be served. Assigning one unit of the demand of location j to center $i \in N$ incurs service costs c_{ij}. We assume the service costs are non-negative, identity of indiscernibles, symmetric and satisfy the triangle inequality. That is, $c_{ij} \geq 0, \forall i, j \in N$; $c_{ij} = 0$, if $i = j$; $c_{ij} = c_{ji}, \forall i, j \in N$ and $c_{it} + c_{tj} \geq c_{ij}, \forall i, j, t \in N$. The objective is to serve all the demands by opening at most k centers and satisfying the capacity constraints such that the total cost is minimized. In this paper, we consider the *hard* capacities and *splittable* demands, that is, we allow at most one center to be opened at any location and each location can be served from more than one open center. (In contrast, the *soft* capacities allows that multiple centers can be opened in a single location. In the *unsplittable* demands case each location must be served by exactly one open center.)

CKM can be formulated as the following mixed integer program (MIP), where variable x_{ij} indicates the fraction of the demand of location j that is served by location i, and y_i indicates whether location i is selected as a center.

$$\begin{aligned} \min \quad & \sum_{i,j \in N} d_j c_{ij} x_{ij} \\ \text{subject to:} \quad & \sum_{i \in N} x_{ij} = 1, \quad \forall j \in N; \quad \sum_{j \in N} d_j x_{ij} \leq M y_i, \quad \forall i \in N; \\ & \sum_{i \in N} y_i \leq k; \qquad\qquad 0 \leq x_{ij} \leq y_i, \quad \forall i, j \in N; \\ & y_i \in \{0, 1\}, \quad \forall i \in N. \end{aligned} \tag{1}$$

Replacing constraints (1) by $0 \leq y_i \leq 1, \forall i \in N$, we obtain the LP-relaxation of CKM.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 325–338

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

1.1 Related Work and Our Results

The k-median problem is a classical NP-hard problem in computer science and operations research, and has a wide variety of applications in clustering and data mining [4, 13]. The uncapacitated k-median problem was studied extensively [1, 2, 6, 8, 9, 14, 15, 17], and the best known approximation algorithm was recently given by Byrka et al. [6] with approximation ratio $2.611 + \epsilon$ by improving the algorithm of Li and Svensson [17].

The capacitated versions of k-median problem are much less understood. The above LP-relaxation has an unbounded integrality gap. More precisely, the capacity or the number of opened centers has to be increased by a factor of at least 2, if we try to get an integral solution within a constant factor of the cost of an optimal solution to the LP-relaxation [9]. All the previous attempts with constant approximation ratios for this problem violate at least one of the two kinds of hard constraints: the capacity constraint and cardinality constraint (at most k centers can be opened), even the local search technique.

For the hard uniform capacity case, by increasing the capacities within a factor of 3, Charikar et al. [7, 9, 12] presented a 16-approximation algorithm based on LP-rounding. This violation ratio of capacities was recently improved to $2 + \frac{3}{l-1}$, $l \in \{2, 3, 4, \dots\}$ by Byrka et al. [5], with the corresponding approximation ratio of $32l^2 + 28l + 7$. In addition, Korupolu et al. [16] proposed a $(1 + 5/\epsilon)$-approximation algorithm while opening at most $(5 + \epsilon)k$ centers, and a $(1 + \epsilon)$-approximation algorithm while opening at most $(5 + 5/\epsilon)k$ centers based on a local search technique.

For soft non-uniform capacities, Chuzhoy and Rabani [10] presented a 40-approximation algorithm while violating the capacities within a factor of 50 based on primal-dual and Lagrangian relaxation methods. Using at most $(1 + \delta)k$ facilities, Bartal et al. [3] gave a $19.3(1 + \delta)/\delta^2$-approximation algorithm ($\delta > 0$). For hard non-uniform capacities, Gijswijt and Li [11] gave a $(7 + \epsilon)$-approximation algorithm while opening at most $2k$ centers.

In this paper, we improve the algorithm of Charikar et al. [9] to reduce its violation ratio of capacities from 3 to $2 + \frac{2}{\alpha}$, $\alpha \geq 4$ and get an $(6+10\alpha)$-approximation algorithm for the hard uniform capacitated k-median problem, which improves the previous best approximation ratio for any violation ratio of capacities in $(2, 3)$. The approximation ratios we obtain for violation ratio of 2.1, 2.3, 2.5, 2.75 and 3 (for instance) are summarized in the following table.

violation ratio of capacities	2.1	2.3	2.5	2.75	3
previous best	31627	4187	1771	947	16
our algorithm	206	72.67	46	46	46

Note that with increasing the capacities by a factor of at least 3, the best approximation ration is still due to Charikar et al. [9].

Additionally, for metric facility location problems there is a slightly different model for the capacitated k-median [5, 11], in which we are given a set F of facilities and a set D of clients. Each facility has a capacity M. Each client $j \in D$ has a demand d_j that has to be served by facilities. Note that the capacity of each client is 0. This is different from our model, in which each location has a capacity M. We show that our algorithm can be easily extended to solve this model with increasing the approximation ratio by a factor at most $2 + \frac{1}{6+10\alpha}$.

1.2 The Main Idea Behind Our Algorithm

In Charikar et al. [9] algorithm, based on an optimal solution to the LP-relaxation, a $\{\frac{1}{2}, 1\}$-solution (x, y) is first constructed such that $y_i \in \{0, \frac{1}{2}, 1\}, \forall i \in N$; $\sum_{j \in N} x_{ij} d_j \leq M$,

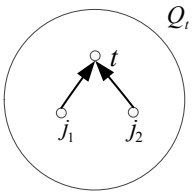


Figure 1 *A star Q_t.*

if $y_i = \frac{1}{2}$; and $\sum_{j\in N} x_{ij}d_j \leq 2M$, if $y_i = 1$. Note that $\sum_{j\in N} x_{ij}d_j \leq My_i$ could be violated in this solution.

Next, a center is directly opened at location i if $y_i = 1$. Then, they construct a collection of rooted stars spanning the locations $i \in N$ with $y_i = \frac{1}{2}$. By a star by star rounding procedure, exactly half of the locations with fractional opening value $\frac{1}{2}$ are chosen as centers. The demands of another half of the locations, where no center is opened finally, are reassigned to the opened half. In the worst case, the capacity of the root of some star has to be increased by factor 3 to satisfy the capacity constraint. Take Fig. 1 as an example. The star Q_t, rooted at t, has two children j_1 and j_2 with $y_t = y_{j_1} = y_{j_2} = \frac{1}{2}$. In the worst case of Charikar et al. algorithm, we are allowed to build at most $\lfloor y_t + y_{j_1} + y_{j_2} \rfloor$ centers, i. e., 1 center. Without loss of generality, suppose we build a center at the root t, and reassign the demand served by j_1 and j_2 to t. Then, the capacity of t has to be increased by factor 3 to satisfy the capacity constraint, as $\sum_{j\in N} x_{ij}d_j \leq M$ for $i = t, j_1, j_2$.

We generalize the algorithm of Charikar et al. to improve its violation ration from 3 to $2 + \epsilon$. The key idea behind our algorithm relies on the following observations. One is that if we can obtain a $\{1 - \frac{1}{\delta}, 1\}$-solution, then 2 centers can be built for the above example in the worst case by setting $\delta \geq 3$, as then $\lfloor y_t + y_{j_1} + y_{j_2} \rfloor \geq \lfloor \frac{2}{3} + \frac{2}{3} + \frac{2}{3} \rfloor = 2$. Consequently, we only need to blow up the capacity of location t by factor 2 instead of 3, by building centers at t and j_2, and assigning the demand served by j_1 to t. However, this example only shows one kind of stars. To make sure the violation ratio can be improved for all kinds of stars, we construct a $\{(\frac{\alpha-2}{\alpha}, \frac{\alpha-1}{\alpha}], [1, 2)\}$-solution (x, y) such that

1. for each $i \in N$, $\frac{\alpha-2}{\alpha} < y_i \leq \frac{\alpha-1}{\alpha}$, or $1 \leq y_i < 2$, or $y_i = 0$; and $|\{i \in N \mid \frac{\alpha-2}{\alpha} < y_i < \frac{\alpha-1}{\alpha}\}| \leq 1$;
2. if $\frac{\alpha-2}{\alpha} < y_i \leq \frac{\alpha-1}{\alpha}$, then $\sum_{j\in N} d_j x_{ij} \leq M$;
3. if $1 \leq y_i < 2$, then $\sum_{j\in N} d_j x_{ij} \leq My_i$.

Another one is that constraints $y_i \leq 1, \forall i \in N$ hold in each step of the algorithm by Charikar et al. That is, they round $y_i > 1$ to be 1 for each $i \in N$ in each step. This is a quite natural operation since we consider the hard capacitated case, i. e., at most one center can be opened at any location. However, we observe that after obtaining an optimal solution to the LP-relaxation, it is sufficient to make sure constraints $y_i \leq 1, \forall i \in N$ hold in our last step. For all other steps (except last step), this rounding can be avoided by relaxing the constraint $y_i \leq 1$ to $y_i < 2$. We use an example to show the profit we can gain from avoiding this rounding. Suppose we have a star Q_t rooted at t with one child j_1. Moreover, $y_t = 1.9$ and $y_{j_1} = 0.5$. Then, in the worst case, we can build $\lfloor y_t + y_{j_1} \rfloor = 2$ centers. We open t and j_1. Consequently, we only need to increase the capacity of t by factor 1.9 (note that if $1 \leq y_i < 2$, then $\sum_{j\in N} d_j x_{ij} \leq My_i$ for our $\{(\frac{\alpha-2}{\alpha}, \frac{\alpha-1}{\alpha}], [1, 2)\}$-solution). However, if we round 1.9 to 1, we obtain a star Q_t with $y_t = 1$ and $y_{j_1} = 0.5$. Then, in the worst case, only 1 center can be built as $\lfloor y_t + y_{j_1} \rfloor = 1$. Without loss of generality, suppose we build a center at t, and assign the demand served by j_1 to t. Then, we need to increase the capacity of t by factor 2.9.

2 An Improved Approximation Algorithm

From now on, let (x, y) denote an optimal solution to the LP-relaxation with total cost C_{LP}. We consider y_i as the *opening value* of location i. If $y_i \in (0, 1)$, we say that location i is *fractionally opened* (as a center). For each $j \in N$, define $C_j = \sum_{i \in N} c_{ij} x_{ij}$. Note that $C_{LP} = \sum_{j \in N} d_j C_j$. The outline of our algorithm is similar to [9].

Step 1. We partition locations to a collection of clusters. The total opening value of each cluster is at least $\frac{\alpha-1}{\alpha}, \alpha \geq 4$.

Step 2. For each cluster, we integrate the nearby opened locations to obtain a $[\frac{\alpha-1}{\alpha}, 2)$-solution (x', y') to the LP-relaxation, which satisfies the relaxing constraints $0 \leq y'_i < 2$ instead of $0 \leq y'_i \leq 1$ for each $i \in N$.

Step 3. We redistribute the opening values among locations with $y'_i \in [\frac{\alpha-1}{\alpha}, 1)$ to obtain a $\{(\frac{\alpha-2}{\alpha}, \frac{\alpha-1}{\alpha}], [1, 2)\}$-solution $(x', \hat{y})$, which satisfies the relaxing constraints $\sum_{j \in N} d_j x'_{ij} \leq M$ if $\hat{y}_i \in (0, 1)$, $\sum_{j \in N} d_j x'_{ij} \leq M \hat{y}_i$ otherwise, instead of $\sum_{j \in N} d_j x'_{ij} \leq M \hat{y}_i$ for each $i \in N$.

Step 4. We round the $\{(\frac{\alpha-2}{\alpha}, \frac{\alpha-1}{\alpha}], [1, 2)\}$-solution to be an integral solution with increasing the capacities by a factor of $2 + \frac{2}{\alpha}$.

2.1 Step 1: Clustering

In this step, by the *filtering* technique of Lin and Vitter [18], we will partition locations into clusters, and for each cluster select a single location as the *core* of this cluster, such that each location in the cluster is not far to its cluster core and the cores are sufficiently far to each other.

Let N' be the collection of all cluster cores. Let $N'(j)$ denote the closest cluster core to j in N'. For each $l \in N'$, let M_l denote the cluster whose core is l, and define $Z_l = \sum_{j \in M_l} y_j$ be the total opening value of all locations in cluster M_l.

▶ **Definition 1.** We call a cluster M_l *terminal* if $Z_l \geq 1$, *non-terminal* if $0 < Z_l < 1$.

Let $n = |N|$. The clustering is done by Procedure 1 (similar to [9]). After this step, the following properties hold ($\alpha \geq 4$):

[**1a**]. $\forall j \in M_l, l \in N'$, $c_{lj} \leq 2\alpha C_j$;

[**1b**]. $\forall l, l' \in N'$ and $l \neq l'$, $c_{ll'} > 2\alpha \max\{C_l, C_{l'}\}$;

[**1c**]. $\forall l \in N', Z_l = \sum_{j \in M_l} y_j \geq \frac{\alpha-1}{\alpha}$;

[**1d**]. $\bigcup_{l \in N'} M_l = N$; and $M_l \bigcap M_{l'} = \emptyset, \forall l, l' \in N'$ and $l \neq l'$.

We can easily get property **1a**, **1b** and **1d** from this procedure.

Note that location i belongs to cluster M_l if $c_{il} \leq \alpha C_l$. For contradiction, suppose for some $i \in N$ with $c_{il} \leq \alpha C_l$, $i \in M_{l'}$ instead of $i \in M_l$, where $l' \in N' - \{l\}$. This means $c_{il'} \leq c_{il}$ as we add i to cluster $M_{l'}$ only if $N'(i) = l'$. Then, we have $c_{ll'} \leq c_{il} + c_{il'} \leq 2c_{il} \leq 2\alpha C_l$, which is a contradiction as $c_{ll'} > 2\alpha C_l$ by property **1b**. Then, we have the following lemma. See [18] for the proof.

▶ **Lemma 2.** (property **1c**) $\forall l \in N', Z_l \geq \frac{\alpha-1}{\alpha}$.

2.2 Step 2: Obtaining a $[\frac{\alpha-1}{\alpha}, 2)$-solution

We will get rid of locations with relatively small fractional opening value in this step, by constructing a $[\frac{\alpha-1}{\alpha}, 2)$-solution (x', y') in which $y'_i = 0$ or $\frac{\alpha-1}{\alpha} \leq y'_i < 2$, $\forall i \in N$. For each cluster M_l, we transfer the amount of locations (their opening values and the demands served by these locations) far away from the cluster core l to locations closer to l.

Procedure 1. Clustering

1. order all locations in nondecreasing order of C_j, (without loss of generality, assume $C_1 \leq \cdots \leq C_n$);
2. set $N' := \emptyset$;
3. **for** $j = 1$ *to* n **do**
   find a location $l \in N'$ such that $c_{lj} \leq 2\alpha C_j$, where $\alpha \geq 4$;
   if *no such location is found* **then**
   choose j as a cluster core, i. e., set $N' := N' \cup \{j\}$;
   end
   end
4. set $M_l := \emptyset$, $\forall l \in N'$;
5. **for** $j = 1$ *to* n **do**
   if j *is closer to cluster core* $l \in N'$ *than all other cluster cores (break ties arbitrarily)* **then**
   add location j to cluster M_l. (i. e., set $M_l := \{j \in N \mid N'(j) = l\}$.)
   end
   end

In this step, initially set $y'_i = y_i, x'_{ij} = x_{ij}, \forall i, j \in N$. Then, we consider clusters one by one. For each cluster $M_l, l \in N'$, order locations in M_l in nondecreasing value of $c_{lj}, j \in M_l$. Without loss of generality, assume we get an order $j_1, \cdots, j_u$ (note that $j_1 = l$). If we decide to move the amount of location j_b to j_a $(1 \leq a < b \leq u)$, then perform Procedure 2 [7, 12].

Procedure 2. Move(j_a,j_b)

1. let $\delta = \min\{1 - y'_{j_a}, y'_{j_b}\}$;
2. for all $j \in N$, set $x'_{j_a j} := x'_{j_a j} + \frac{\delta}{y'_{j_b}} x'_{j_b j}$, $x'_{j_b j} := x'_{j_b j} - \frac{\delta}{y'_{j_b}} x'_{j_b j}$;
3. set $y'_{j_a} := y'_{j_a} + \delta$,$y'_{j_b} := y'_{j_b} - \delta$;

▶ **Lemma 3.** *After Procedure 2, we still have*

1. $\sum_{j \in M_l} y'_j = \sum_{j \in M_l} y_j$, *for each* $l \in N'$;
2. *for each* $j \in N$, $\sum_{i \in N} x'_{ij} = 1$;
3. $\sum_{j \in N} d_j x'_{ij} \leq M y'_i$, *for each* $i \in N$.

We use Procedure 3 to decide whether we move the amount of location j_b to j_a.

▶ **Lemma 4.** *If in Procedure 3,* j_a *exists but* j_b *does not exist, and* M_l *is a terminal cluster, then* $a \geq 2$ *and* $y'_{j_{a-1}} = 1$.

Proof. Since M_l is a terminal cluster, we have $Z_l \geq 1$. Moreover, we know $y'_{j_t} = 1$ for each $t < a$ and $y'_{j_s} = 0$ for each $s > a$, as j_b does not exist. Thus, $a \geq 2$. Otherwise, $Z_l < 1$, a contradiction. ◀

▶ **Lemma 5.** *After this step, we have the following properties*

[**2a**]. *for all* $i \in N$, $\frac{\alpha-1}{\alpha} \leq y'_i < 2$ *or* $y'_i = 0$; *and* $\sum_{j \in N} d_j x'_{ij} \leq M y'_i$;
[**2b**]. $\sum_{i \in N} y'_i = \sum_{i \in N} y_i \leq k$;
[**2c**]. $x'_{ij} \leq y'_i, \forall i, j \in N$.

Procedure 3. Concentrate(M_l)

while *there exists a location in* M_l *with fractional opening value* **do**
1. let j_a be the first location in the sequence $j_1, \cdots, j_u$ such that $0 \leq y'_{j_a} < 1$;
2. let j_b be the first location in the sequence $j_{a+1}, \cdots, j_u$ such that $0 < y'_{j_b} \leq 1$;
3. **if** j_a *and* j_b *both exist* **then**
   execute procedure Move(j_a,j_b) to move the amount of j_b to j_a;
end
4. **if** j_a *exists but* j_b *does not exist* **then**
   if M_l *is a terminal cluster,i. e.,*$a \geq 2$ **then**
      set $y'_{j_{a-1}} := y'_{j_{a-1}} + y'_{j_a}, y'_{j_a} := 0$;
      for each $j \in N$, set $x'_{j_{a-1}j} := x'_{j_{a-1}j} + x'_{j_aj}, x'_{j_aj} := 0$;
   end
   terminate.
end
end

Proof. Property **2a.** If M_l is a non-terminal cluster, i. e., $0 < Z_l < 1$, then we will move the amount of each location in M_l to its core l according to Procedure 3. Consequently, we obtain $\frac{\alpha-1}{\alpha} \leq y'_l = Z_l < 1$ (property **1c**) and $y'_j = 0, \forall j \in M_l - \{l\}$.

If M_l is a terminal cluster, i. e., $Z_l \geq 1$, then according to Lemma 4 we get $y'_{j_t} = 1$ for each $t < a$ and $y'_{j_s} = 0$ for each $s > a$ if j_a exists and j_b does not exist. Then, we move the amount of y'_{j_a} to $y'_{j_{a-1}}$. So, $1 \leq y'_{j_{a-1}} < 2$ as $0 \leq y'_{j_a} < 1$. Note that if j_a does not exist, we know $y'_j = 1$ for each $j \in M_l$.

Thus, for all $i \in N$, $\frac{\alpha-1}{\alpha} \leq y'_i < 2$ or $y'_i = 0$. $\sum_{j \in N} d_j x'_{ij} \leq M y'_i, \forall i \in N$ hold by Lemma 3 (note that it is easy to check these inequalities still hold after the step 4 in Procedure 3).

Property **2b.** This directly follows by Lemma 3(1).

Property **2c.** Initially, we set $y'_i = y_i, x'_{ij} = x_{ij}$ for all $i, j \in N$. Thus, $x'_{ij} \leq y'_i$ holds, for each $i, j \in N$. We will show that after the procedure these inequalities still hold.

For each non-terminal cluster, only the core has a positive opening value after this step. And in the procedure the opening value of core is always increased by a bigger amount than the increasing of the fraction of the demand served by the core.

For a terminal cluster, each location i in the cluster has $y'_i = 0$ or $y'_i \geq 1$ after this step. Note that for each location $i \in N$ with $y'_i \geq 1$, $x'_{ij} \leq y'_i$ holds for each $j \in N$ as $x'_{ij} \leq 1$. Moreover, observe that for each $j \in N$, we always set $x'_{ij} := 0$ if y'_i is already set to be 0. ◀

Since each location is not far away from its cluster core, these transfer operations would not increase too much extra cost. More precisely, we can bound the service cost by the following lemma. The proof is similar as Lemma 2.8.3 and 2.8.3 in [7].

▶ **Lemma 6.** *(1). Let* M_l *be a non-terminal cluster. The demand of location* j *originally served by* $j_b (j_b \in M_l)$ *must be served by core* l *after the procedure. And we have* $c_{lj} \leq 2c_{j_bj} + 2\alpha C_j$.

(2). Let M_l *be a terminal cluster. If we move the demand of location* j *served by* j_b *to* j_a $(j_a, j_b \in M_l,\ a < b)$, *we have* $c_{j_aj} \leq 3c_{j_bj} + 4\alpha C_j$.

Let $N_1 = \{i \in N \mid y'_i \geq 1\}$ be the collection of locations with the opening value at least 1. Let $N_2 = \{i \in N \mid y'_i \in [\frac{\alpha-1}{\alpha}, 1)\}$ be the collection of locations with fractional opening value

in $[\frac{\alpha-1}{\alpha}, 1)$. Note that N_2 can also be written as $\{i \in N' \mid Z_i \in [\frac{\alpha-1}{\alpha}, 1)\}$. That is, N_2 is the collection of non-terminal cluster cores. Moreover, we have $N_1 \cup N_2 \supseteq N'$.

▶ **Lemma 7.** *If $|N_2| - 1 < \sum_{i\in N_2} y'_i$, we can get an integer solution with increasing the capacity by factor 2, by opening all locations in $N_1 \cup N_2$ as centers. The total cost of the obtained solution can be bounded by $(3 + 4\alpha)C_{LP}$.*

Proof. If $|N_2|-1 < \sum_{i\in N_2} y'_i$, then $|N_2| = \lceil \sum_{i\in N_2} y'_i \rceil$ as $y'_i < 1$ for each $i \in N_2$. Additionally, since $\sum_{i\in N_1} y'_i \le k - \sum_{i\in N_2} y'_i$ (by property **2b**) and $y'_i \ge 1$ for each $i \in N_1$, we have $|N_1| \le \lfloor k - \sum_{i\in N_2} y'_i \rfloor$.

Thus, if we only open locations in $N_1 \cup N_2$, then we open at most k centers as $\lceil \sum_{i\in N_2} y'_i \rceil + \lfloor k - \sum_{i\in N_2} y'_i \rfloor = k$.

Since $y'_i = 0$ for each $i \notin N_1 \cup N_2$, we have $\sum_{i\in N_1\cup N_2} x'_{ij} = 1, \forall j \in N$ by Lemma 3(2) and property **2c.** That is, $\sum_{i\in N_1\cup N_2} d_j x'_{ij} = d_j$ for each $j \in N$. Thus, the demand of each $j \in N$ can be satisfied by assigning $d_j x'_{ij}$ to $i \in N_1 \cup N_2$.

By Lemma 6, it is easy to see that the total cost of the obtained solution can be bounded by $(3 + 4\alpha)C_{LP}$. By Lemma 5, we know for all $i \in N$, $\frac{\alpha-1}{\alpha} \le y'_i < 2$ or $y'_i = 0$; and $\sum_{j\in N} d_j x'_{ij} \le M y'_i$. So, we increase the capacity by at most a factor of 2. ◀

From now on, we only consider the following case.

▶ **Assumption 8.** $\sum_{i\in N_2} y'_i \le |N_2| - 1$.

▶ **Definition 9.** We define new demands d' as follows. For each $i \in N$, set $d'_i := \sum_{j\in N} d_j x'_{ij}$. (Note that $d'_i = 0$ for each $i \in N - (N_1 \cup N_2)$.)

2.3 Step 3: Obtaining a $\{(\frac{\alpha-2}{\alpha}, \frac{\alpha-1}{\alpha}], [1, 2)\}$-solution

For each $i \in N_2$, let $s(i)$ be the nearest location to i in $(N_1 \cup N_2) - \{i\}$ (break ties arbitrarily). Let $Y = \sum_{i\in N_2} y'_i$. Note that we only consider the case: $Y \le |N_2| - 1$ by Assumption 8. After this step we will obtain a solution $(x', \hat{y})$ with $\frac{\alpha-2}{\alpha} < \hat{y}_i \le \frac{\alpha-1}{\alpha}$, or $1 \le \hat{y}_i < 2$, or $\hat{y}_i = 0$ for each $i \in N$.

In this step, initially we order all locations in N_2 in nondecreasing order of $d'_i c_{s(i)i}$. Without loss of generality, suppose we get an order $i_1, \cdots, i_v$. Next, for each $i \in N - N_2$, set $\hat{y}_i := y'_i$. For each $i \in N_2$, set $\hat{y}_i := \frac{\alpha-1}{\alpha}$. Let $Y' := Y - \sum_{i\in N_2} \hat{y}_i$. Then, perform Procedure 4.

▶ Remark. The Procedure 4 terminates at $r > 1$. If the procedure terminates at $r = 1$, then we get $Y = \sum_{t=1}^{v} y'_{i_t} > |N_2| - 1$, a contradiction.

▶ **Lemma 10.** *After the above procedure, we have the following properties*

[**3a**]. *for all $i \in N$, $\frac{\alpha-2}{\alpha} < \hat{y}_i \le \frac{\alpha-1}{\alpha}$, or $1 \le \hat{y}_i < 2$, or $\hat{y}_i = 0$; and only $\hat{y}_{i_1}$ can be in $(\frac{\alpha-2}{\alpha}, \frac{\alpha-1}{\alpha})$, i. e., $|\{i \in N \mid \frac{\alpha-2}{\alpha} < \hat{y}_i < \frac{\alpha-1}{\alpha}\}| \le 1$;*

[**3b**]. *for any location $i \in N$, if $\frac{\alpha-2}{\alpha} < \hat{y}_i \le \frac{\alpha-1}{\alpha}$, then $d'_i = \sum_{j\in N} d_j x'_{ij} \le M$;*

[**3c**]. *for any location $i \in N$, if $1 \le \hat{y}_i < 2$, then $d'_i = \sum_{j\in N} d_j x'_{ij} \le M\hat{y}_i$;*

[**3d**]. $\sum_{i\in N_2} \hat{y}_i = \sum_{i\in N_2} y'_i$; $\sum_{i\in N} \hat{y}_i = \sum_{i\in N} y'_i \le k$;

[**3e**]. $\sum_{i\in N_2} (1 - \hat{y}_i) d'_i c_{s(i)i} \le \sum_{i\in N_2} (1 - y'_i) d'_i c_{s(i)i}$.

Proof. Property **3a**. For each location $i \in N - N_2$, we set $\hat{y}_i := y'_i$. So, $1 \le \hat{y}_i < 2$ for each $i \in N_1$; $\hat{y}_i = 0$ for each $i \in N - (N_1 \cup N_2)$.

For each location $i \in N_2$, initially we set $\hat{y}_i := \frac{\alpha-1}{\alpha}$. In the Procedure 4, only $\hat{y}_{i_1}$ could be decreased by a number in $(0, \frac{1}{\alpha})$. The opening value of other location in N_2 remains the same or is set to be 1.

Procedure 4. Determine new opening values for $N_2(Y \leq |N_2| - 1)$

for $r = v$ *to* 1 **do**
  if $Y' = 0$ **then**
    terminate;
  end
  if $Y' > 0$ *and* $Y' + \hat{y}_{i_r} < 1$ **then**
    set $\hat{y}_{i_1} := \hat{y}_{i_1} - (1 - Y' - \hat{y}_{i_r}), \hat{y}_{i_r} := 1$;
    terminate;
  end
  if $Y' > 0$ *and* $Y' + \hat{y}_{i_r} \geq 1$ **then**
    set $\hat{y}_{i_r} := 1$ and update $Y' := Y - \sum_{i \in N_2} \hat{y}_i$;
  end
end

Property **3b, 3C**. Notice that if for location i we have $\frac{\alpha-2}{\alpha} < \hat{y}_i \leq \frac{\alpha-1}{\alpha}$ after the procedure, then we know $\frac{\alpha-1}{\alpha} \leq y'_i < 1$. And if $1 \leq \hat{y}_i < 2$ for location i after the procedure, then we have $y'_i \leq \hat{y}_i$.

We make no change on x'. Thus, combining with property **2a**, we have if $\frac{\alpha-2}{\alpha} < \hat{y}_i \leq \frac{\alpha-1}{\alpha}$, then $\sum_{j \in N} d_j x'_{ij} \leq M y'_i < M$. If $1 \leq \hat{y}_i < 2$, then $\sum_{j \in N} d_j x'_{ij} \leq M y'_i \leq M \hat{y}_i$.

Property **3d**. We move the opening value from one location to the other locations. We do not change the total opening value. So, $\sum_{i \in N_2} \hat{y}_i = \sum_{i \in N_2} y'_i$ holds after Procedure 4. Moreover, we set $\hat{y}_i := y'_i$ for each $i \in N - N_2$. Thus, we also have $\sum_{i \in N} \hat{y}_i = \sum_{i \in N} y'_i \leq k$.

Property **3e**. We always transfer the opening value from i_a to i_b, where $a < b$ and $d'_{i_b} c_{s(i_b)i_b} \geq d'_{i_a} c_{s(i_a)i_a}$. Therefore, $\sum_{i \in N_2} \hat{y}_i d'_i c_{s(i)i} \geq \sum_{i \in N_2} y'_i d'_i c_{s(i)i}$. Then, we have $\sum_{i \in N_2} (1 - \hat{y}_i) d'_i c_{s(i)i} \leq \sum_{i \in N_2} (1 - y'_i) d'_i c_{s(i)i}$. ◀

2.4 Step 4: Rounding to an Integral Solution

Let $\hat{N}_1 = \{i \in N \mid 2 > \hat{y}_i \geq 1\}$ be the set of locations with opening value greater than or equal to 1. Let $\hat{N}_2 = \{i \in N \mid \frac{\alpha-2}{\alpha} < \hat{y}_i \leq \frac{\alpha-1}{\alpha}\}$ be the set of location with fractional opening value strictly less than 1. Let $L_1 = |\hat{N}_1|$. Note that $N_1 \cup N_2 = \hat{N}_1 \cup \hat{N}_2$, and $\hat{N}_2 \subseteq N_2$.

In this step, we aim to construct an integral solution $(\bar{x}, \bar{y})$ with $\sum_{j \in N} \bar{x}_{ij} d'_j \leq (2 + \frac{2}{\alpha}) M \bar{y}_i$ for each $i \in N$. If location j is opened as a center, we serve the demand d'_j of location j by itself. That is, set $\bar{x}_{jj} := 1, \bar{x}_{ij} := 0$ for each $i \neq j, i \in N$. And we build a center at location i if $1 \leq \hat{y}_i < 2$, i. e., set $\bar{y}_i := 1$ for each $i \in \hat{N}_1$. For $\hat{N}_2$, we will open at most $k - L_1$ locations as centers. If a center is not opened at location $j \in \hat{N}_2$, we assign the demand d'_j of j to another opened center i, i. e., set $\bar{x}_{ij} := 1$. Now we start to show the details of this step.

Initially, for each $i, j \in N$ set $\bar{x}_{ij} := 0$; and $\bar{y}_i := 0$. Then, we construct a collection of rooted trees spanning the locations in $\hat{N}_2$ as in [9]. Recall that $s(i)$ is the closest location to i in $(\hat{N}_1 \cup \hat{N}_2) - \{i\}$ $(N_1 \cup N_2 = \hat{N}_1 \cup \hat{N}_2)$ for each $i \in N_2$. We draw a directed edge from i to $s(i)$ if $i \in \hat{N}_2$. The cycles can be eliminated by the following way. For each cycle, we take any location in this cycle as a root and delete the edge from this root to other location. If there is a directed edge from i to $s(i)$ finally, we consider $s(i)$ as the parent of i. Then, we get a desired collection of rooted trees.

Next, we decompose each tree into a collection of rooted stars by Procedure 5.

▶ Remark. In each rooted star, all the children of the root have a fractional opening value. If the root of a star is a fractionally opened location, then the root has at least one child.

Procedure 5. Decompose a tree T to stars

while *there are at least two nodes in* T **do**
  choose a leaf node i with biggest number of edges on the path from i to the root;
  consider the subtree rooted at $s(i)$ as a rooted star, and remove this subtree;
end
if *only one node* i *is left and* $0 < \hat{y}_i < 1$ **then**
  add i to the star rooted at $s(i)$ as a child of $s(i)$;
end

▶ **Definition 11.** An even star is a star with even number of children. An odd star is a star with odd number of children.

Let Q_t denote the star rooted at location t. By abuse of notation, we also use Q_t to denote the collection of locations in the star rooted at t. Let $R_t = \sum_{i \in Q_t} \hat{y}_i$ be the total opening value in Q_t.

▶ **Lemma 12.** **(1)** *If a star* Q_t *has even positive number of fractionally opened locations, i. e.,* $|Q_t \cap \hat{N}_2| = 2q$ *is an even number and* $q \in \mathbb{Z}^+$*, then the total opening value of these fractionally opened locations is greater than* q*, i. e.,* $\sum_{i \in Q_t \cap \hat{N}_2} \hat{y}_i > q$.
(2) *If* $|Q_t \cap \hat{N}_2| = 2q+1$ *is an odd number and* $q \in \mathbb{Z}^+$*, then* $\sum_{i \in Q_t \cap \hat{N}_2} \hat{y}_i > q+1$.

Proof. **(1)** By property **3a**, $|\{i \in N \mid \frac{\alpha-2}{\alpha} < \hat{y}_i < \frac{\alpha-1}{\alpha}\}| \le 1$. So in $\hat{N}_2$ at most one location has a fractional opening value in $(\frac{\alpha-2}{\alpha}, \frac{\alpha-1}{\alpha})$, and all other locations have fractional opening value exactly equal to $\frac{\alpha-1}{\alpha}$.
So,

$$\sum_{i \in Q_t \cap \hat{N}_2} \hat{y}_i > \frac{\alpha-2}{\alpha} + \frac{\alpha-1}{\alpha}(2q-1) = \frac{2q\alpha - 2q - 1}{\alpha} = q + \frac{q\alpha - 2q - 1}{\alpha}.$$

Moreover, since $\alpha \ge 4$ and $q \ge 1$, we have $\frac{q\alpha-2q-1}{\alpha} \ge \frac{2q-1}{\alpha} > 0$. Thus, $\sum_{i \in Q_t \cap \hat{N}_2} \hat{y}_i > q$.
(2) First, we have

$$\sum_{i \in Q_t \cap \hat{N}_2} \hat{y}_i > \frac{\alpha-2}{\alpha} + \frac{\alpha-1}{\alpha}2q = \frac{2q\alpha - 2q + \alpha - 2}{\alpha} = q + 1 + \frac{q\alpha - 2q - 2}{\alpha}.$$

Then, as $\alpha \ge 4$ and $q \ge 1$, we get $\frac{q\alpha-2q-2}{\alpha} \ge \frac{2q-2}{\alpha} \ge 0$. Thus, $\sum_{i \in Q_t \cap \hat{N}_2} \hat{y}_i > q+1$. ◀

We build a center at each location $i \in \hat{N}_1 - \bigcup_t Q_t$ (locations are in $\hat{N}_1$, but not in any star), i. e., set $\bar{y}_i := 1$ and $\bar{x}_{ii} := 1$. For each kind of star Q_t, we define operations to make sure at most $\lfloor R_t \rfloor$ locations in Q_t are selected to be centers.

1. **An even star rooted at location** t **with** $1 \le \hat{y}_t < 2$**.** Let $i_1, \cdots, i_{2q}$ be a sequence of all its children in nondecreasing order of distance from t. We build centers at location $t, i_1, i_3, \cdots, i_{2q-1}$, and serve the demand $d'_{i_{2r}}$ of i_{2r} by opened location i_{2r-1}, i. e.,

$$\text{set } \bar{y}_t := 1; \quad \bar{y}_{i_{2r-1}} := 1, \bar{y}_{i_{2r}} := 0, r = 1, \cdots, q;$$
$$\text{set } \bar{x}_{tt} := 1; \quad \bar{x}_{i_{2r-1}i_{2r-1}} := 1, \bar{x}_{i_{2r-1}i_{2r}} := 1, r = 1, \cdots, q.$$

2. **An even star rooted at location t with $\frac{\alpha-2}{\alpha} < \hat{y}_t \leq \frac{\alpha-1}{\alpha}$.** Let $i_1, \cdots, i_{2q}$ be a sequence of all its children in nondecreasing order of distance from t. (Note that $q \geq 1$ by the before Remark.) We build centers at location $t, i_2, i_4, \cdots, i_{2q}$, and serve the demand $d'_{i_{2r+1}}$ of i_{2r+1} by opened location i_{2r}, serve the demand d'_{i_1} of i_1 by t.
3. **An odd star rooted at location t with $1 + \frac{2}{\alpha} \leq \hat{y}_t < 2$.** Let $i_1, \cdots, i_{2q+1}$ be a sequence of all its children in nondecreasing order of distance from t. We open $t, i_1, i_3, \cdots, i_{2q+1}$ as centers, and serve the demand $d'_{i_{2r}}$ of i_{2r} by opened location i_{2r-1}.
4. **An odd star rooted at location t with $\frac{\alpha-2}{\alpha} < \hat{y}_t \leq \frac{\alpha-1}{\alpha}$ or $1 \leq \hat{y}_t < 1 + \frac{2}{\alpha}$.** Let $i_1, \cdots, i_{2q+1}$ be a sequence of all its children in nondecreasing order of distance from t. We build centers at location $t, i_2, i_4, \cdots, i_{2q}$, and serve the demand $d'_{i_{2r+1}}$ of i_{2r+1} by opened location i_{2r}, serve the demand d'_{i_1} of i_1 by t.

Note that $(\bar{x}, \bar{y})$ is an integral solution for new demands d'. To get an integral solution for our original demands d, we can redistribute the demands d' to their original locations according to Definition 9.

3 Analysis

By property **3a**, **3b** and **3c**, and Lemma 12, we can get the following lemma.

▶ **Lemma 13.** *For each kind of star Q_t, we build at most $\lfloor R_t \rfloor$ centers. And for each $i \in N$, we have $\sum_{j \in N} d'_j \bar{x}_{ij} \leq (2 + \frac{2}{\alpha}) M \bar{y}_i$.*

▶ **Lemma 14.** *We build at most k centers, and increase capacities by factor $2 + \frac{2}{\alpha}$.*

Proof. Suppose we get stars $Q_1, \cdots, Q_t$ by decomposing all the trees in Step 4. Then by property **3d**, we know $\sum_{r=1}^{t} R_r + \sum_{i \in \hat{N}_1 - \bigcup_{r=1}^{t} Q_r} \hat{y}_i \leq k$. Moreover, we build at most $\sum_{r=1}^{t} \lfloor R_r \rfloor + \sum_{i \in \hat{N}_1 - \bigcup_{r=1}^{t} Q_r} \lfloor \hat{y}_i \rfloor$ centers by Lemma 13 and the operation for locations that are in $\hat{N}_1$ but not in any star. Consequently, we build at most k centers. Again, by Lemma 13 we increase the capacity by at most a factor of $2 + \frac{2}{\alpha}$ to satisfy all the demand constraints. ◀

For each location i in star Q_t, let $r(i) \in Q_t$ denote the location that the demand d'_i of i is reassigned to. Define the cost of star Q_t as $\sum_{i \in Q_t} d'_i c_{r(i)i}$.

▶ **Lemma 15.** *The cost of stars can be bounded by $\sum_{i \in N_2} \sum_{j \in N} \sum_{i' \in M_i} d_j (4 c_{i'j} x_{i'j} + 8\alpha C_j x_{i'j})$.*

Proof. Note that in this proof we only consider location $i \in \hat{N}_2$, since we always build a center at each location in $\hat{N}_1$ and serve its demand by itself.

For each star Q_t, the reassignment is always to serve the demand d'_i of location i by an opened location i' that is closer to the root t, where $i, i' \in Q_t$ and $c_{ti'} \leq c_{ti}$. Recall that $s(i)$ is the closest location to i in $(N_1 \cup N_2) - \{i\}$. By Procedure 5, we know $s(i) = s(i') = t$. The cost for this reassignment is $d'_i c_{i'i}$, which can be bounded by $2d'_i c_{s(i)i}$ as $c_{i'i} \leq c_{s(i)i'} + c_{s(i)i} \leq 2c_{s(i)i}$.

Since $\frac{\alpha-2}{\alpha} < \hat{y}_i \leq \frac{\alpha-1}{\alpha}$ for each $i \in Q_t \cap \hat{N}_2$, we have $2d'_i c_{s(i)i} \leq 2\alpha(1 - \hat{y}_i) d'_i c_{s(i)i}$.

We sum $2\alpha(1 - \hat{y}_i) d'_i c_{s(i)i}$ over all $i \in \hat{N}_2$ to get an upper bound for the total cost of stars, i. e., $\sum_{i \in \hat{N}_2} 2\alpha(1 - \hat{y}_i) d'_i c_{s(i)i}$. Note that $\hat{N}_2 \subseteq N_2$. Then, by property **3e**,the definition of d'_i and Procedure 3 (Lemma 6), we know

$$\sum_{i \in \hat{N}_2} 2\alpha(1 - \hat{y}_i) d'_i c_{s(i)i} \leq \sum_{i \in N_2} \sum_{j \in N} \sum_{i' \in M_i} 2\alpha(1 - y'_i) d_j x_{i'j} c_{s(i)i}.$$

Therefore, it is sufficient to show that for each $j \in N, i' \in M_i, i \in N_2$

$$2\alpha(1-y'_i)d_j x_{i'j} c_{s(i)i} \le d_j(4c_{i'j}x_{i'j} + 8\alpha C_j x_{i'j}).$$

We have two cases: (a) $N'(j) = i$ and (b) $N'(j) \neq i$. We show the above inequality holds for both cases.

(a) $N'(j) = i$.

Since $y'_i \in [\frac{\alpha-1}{\alpha}, 1), \forall i \in N_2$, we can find a location $i^* \notin M_i$ with $x_{i^*i} > 0$ and $c_{i^*i} \le \frac{C_i}{1-y'_i}$. Otherwise, $\sum_{r\in N} x_{ri}c_{ri} > C_i$, a contradiction.

Note that $c_{N'(i^*)i^*} \le c_{ii^*}$ since $N'(i^*) \neq i$, and $N'(i^*)$ is the closest location to i^* in N', and $i \in N'$. So, $c_{s(i)i} \le c_{iN'(i^*)} \le c_{N'(i^*)i^*} + c_{ii^*} \le 2c_{ii^*} \le 2\frac{C_i}{1-y'_i}$.

If $C_i \le C_j$, then we have

$$2\alpha(1-y'_i)d_j x_{i'j} c_{s(i)i} \le 2\alpha d_j x_{i'j} 2C_i \le 4\alpha d_j x_{i'j} C_j. \tag{2}$$

Otherwise $C_i > C_j$. Then, we consider location j before i when we choose the cluster cores N', and j can not be a cluster core. This means there exists a location $r \in N'$ with $C_r \le C_j$ and $C_{rj} \le 2\alpha C_j$ before we check whether j should be chosen as a cluster core. So, $2\alpha C_i < c_{ri} \le c_{rj} + c_{ij} \le 2\alpha C_j + 2\alpha C_j = 4\alpha C_j$. That is, $C_i \le 2C_j$. Thus, for this case we have

$$2\alpha(1-y'_i)d_j x_{i'j} c_{s(i)i} \le 2\alpha d_j x_{i'j} 2C_i \le 8\alpha d_j x_{i'j} C_j. \tag{3}$$

(b) $N'(j) \neq i$.

The proof for this case is similar as that in [7, 12]. First, we have

$$c_{s(i)i} \le c_{N'(j)i} \le c_{i'i} + c_{i'N'(j)} \le 2c_{i'N'(j)} \le 2(c_{i'j} + c_{N'(j)j}),$$

where $i' \in M_i$.

By property **1a**, $c_{N'(j)j} \le 2\alpha C_j$. So, $c_{s(i)i} \le 2c_{i'j} + 4\alpha C_j$.

Note that $0 < \alpha(1-y'_i) \le 1$ as $1 > y'_i \ge \frac{\alpha-1}{\alpha}, i \in N_2$. Thus, we have

$$2\alpha(1-y'_i)d_j x_{i'j} c_{s(i)i} \le 2d_j x_{i'j}(2c_{i'j} + 4\alpha C_j) = d_j(4c_{i'j}x_{i'j} + 8\alpha C_j x_{i'j}). \tag{4}$$

From inequalities (2), (3) and (4), we get

$$2\alpha(1-y'_i)d_j x_{i'j} c_{s(i)i} \le d_j(4c_{i'j}x_{i'j} + 8\alpha C_j x_{i'j}).$$

◀

In our algorithm, we reassign the service twice: in Step 2 and Step 4. The cost of reassignment for Step 2 (Step 4) can be bounded by Lemma 6 (Lemma 15). Combining these two upper bounds, the total cost can be bounded by

$$\begin{aligned}
&\sum_{i\in N_2}\sum_{j\in N}\sum_{i'\in M_i} d_j(2c_{i'j} + 2\alpha C_j)x_{i'j} + \sum_{i\in N'-N_2}\sum_{j\in N}\sum_{i'\in M_i} d_j(3c_{i'j} + 4\alpha C_j)x_{i'j} \\
&+ \sum_{i\in N_2}\sum_{j\in N}\sum_{i'\in M_i} d_j(4c_{i'j}x_{i'j} + 8\alpha C_j x_{i'j}) \\
&\le \sum_{i\in N}\sum_{j\in N} d_j(6c_{ij} + 10\alpha C_j)x_{ij} = \sum_{j\in N} d_j(6C_j + 10\alpha C_j) = (6+10\alpha)C_{LP}.
\end{aligned}$$

Then combining with Lemma 7 and 14, we can prove the following theorem.

▶ **Theorem 16.** *For any $\alpha \ge 4$, there is a $(6+10\alpha)$-approximation algorithm for the hard uniform capacitated k-median problem with increasing the capacity by factor at most $2 + \frac{2}{\alpha}$.*

4 Extent Our Algorithm to Solve Another Model

As mentioned in the introduction, the following model is also considered in some references for the capacitated k-median problem, where variable x_{ij} indicates the fraction of the demand of client j that is served by facility i, and y_i indicates if facility i is open. Let y_i take value one if facility i is open and value zero otherwise. We denote this model by *CKL*.

$$\begin{aligned} \min \quad & \sum_{i\in F}\sum_{j\in D} d_j c_{ij} x_{ij} \\ \text{subject to:} \quad & \sum_{i\in F} x_{ij} = 1, \quad \forall j \in D; \qquad \sum_{j\in D} d_j x_{ij} \le M y_i, \quad \forall i \in F; \\ & \sum_{i\in F} y_i \le k; \qquad\qquad 0 \le x_{ij} \le y_i, \quad \forall i \in F, j \in D, \\ & y_i \in \{0,1\}, \quad \forall i \in F. \end{aligned} \tag{5}$$

Replacing constraints (5) by $0 \le y_i \le 1, i \in F$, we get the LP-relaxation of CKL.

4.1 The Algorithm

Let (x^0, y^0) be an optimal solution to the LP-relaxation of CKL. For each facility $i \in F$, define a demand

$$d_i^1 = \sum_{j\in D} d_j x_{ij}^0.$$

To make use of the algorithm presented in Section 2, we set $N := F$. That is, each location $i \in N$ has a capacity M and demand d_i^1. Then, we get an instance of CKM considered in Section 2. Suppose we get an integral solution (x^1, y^1) for this constructed instance by the algorithm proposed in Section 2.

Then, we construct an integral solution (x^*, y^*) for the original instance of CKL by redistributing the demands $d_{i'}^1$ of location (facility) $i' \in N$ back to clients D. That is, set $y^* := y^1$; and set $x_{ij}^* := \sum_{i'\in N}(x_{ii'}^1 x_{i'j}^0)$, for each $i \in N = F, j \in D$.

4.2 Analysis

We only blow up the capacity once at the moment when we use the algorithm proposed in Section 2 to resolve the constructed instance. Theorem 16 states that this violation ratio is at most $2 + \frac{2}{\alpha}$. Thus, we have the following result.

▶ **Lemma 17.** *(x^*, y^*) is an integral solution for CKL with $\sum_{j\in D} d_j x_{ij}^* \le (2 + \frac{2}{\alpha}) M y_i^*$ for each $i \in F$, where $\alpha \ge 4$.*

▶ **Lemma 18.** *For any $\alpha \ge 4$, there is a $(13 + 20\alpha)$-approximation algorithm for CKL by increasing the capacity by factor $2 + \frac{2}{\alpha}$.*

Proof. Let $COST(\cdot,\cdot)$ be the total cost of solution $(\cdot,\cdot)$. Let OPT_{CKL} and OPT_{CKM} be the optimal objective value of our original instance and constructed instance respectively.

By the process to obtain the constructed instance, we have $OPT_{CKM} \le OPT_{CKL} + COST(x^0, y^0)$. Then,

$$\begin{aligned} & COST(x^*, y^*) \\ & \le COST(x^1, y^1) + COST(x^0, y^0) \le (6 + 10\alpha) OPT_{CKM} + COST(x^0, y^0) \\ & \le (6 + 10\alpha)(OPT_{CKL} + COST(x^0, y^0)) + COST(x^0, y^0) \le (13 + 20\alpha) OPT_{CKL}, \end{aligned}$$

where the first inequality holds according to the process to get the solution (x^*, y^*) and triangle inequalities; the second inequality follows by Theorem 16; the last inequality holds as $COST(x^0, y^0) \leq OPT_{CKL}$. ◀

Acknowledgements. We thank Dion Gijswijt for insightful discussions.

References

1 Aaron Archer, Ranjithkumar Rajagopalan, and David B. Shmoys. Lagrangian relaxation for the k-median problem: New insights and continuity properties. In Giuseppe Di Battista and Uri Zwick, editors, *ESA*, volume 2832 of *LNCS*, pages 31–42. Springer, 2003.

2 Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristic for k-median and facility location problems. In Jeffrey Scott Vitter, Paul G. Spirakis, and Mihalis Yannakakis, editors, *STOC*, pages 21–29. ACM, 2001.

3 Yair Bartal, Moses Charikar, and Danny Raz. Approximating min-sum k-clustering in metric spaces. In Jeffrey Scott Vitter, Paul G. Spirakis, and Mihalis Yannakakis, editors, *STOC*, pages 11–20. ACM, 2001.

4 Paul S. Bradley, Usama M. Fayyad, and Olvi L. Mangasarian. Mathematical programming for data mining: Formulations and challenges. *INFORMS Journal on Computing*, 11(3):217–238, 1999.

5 Jaroslaw Byrka, Krzysztof Fleszar, Bartosz Rybicki, and Joachim Spoerhase. A constant-factor approximation algorithm for uniform hard capacitated k-median. *CoRR*, abs/1312.6550, 2013.

6 Jaroslaw Byrka, Thomas Pensyl, Bartosz Rybicki, Aravind Srinivasan, and Khoa Trinh. An improved approximation for k-median, and positive correlation in budgeted optimization. *CoRR*, abs/1406.2951, 2014.

7 Moses Charikar. *Algorithms for clustering problems.* PhD thesis, Standford University, 2000.

8 Moses Charikar and Sudipto Guha. Improved combinatorial algorithms for the facility location and k-median problems. In *FOCS*, pages 378–388. IEEE Computer Society, 1999.

9 Moses Charikar, Sudipto Guha, Éva Tardos, and David B. Shmoys. A constant-factor approximation algorithm for the k-median problem (extended abstract). In Jeffrey Scott Vitter, Lawrence L. Larmore, and Frank Thomson Leighton, editors, *STOC*, pages 1–10. ACM, 1999.

10 Julia Chuzhoy and Yuval Rabani. Approximating k-median with non-uniform capacities. In *SODA*, pages 952–958. SIAM, 2005.

11 Dion Gijswijt and Shanfei Li. Approximation algorithms for the capacitated k-facility location problems. *CoRR*, abs/1311.4759, 2013.

12 Sudipto Guha. *Approximation algorithm for faciity location problems.* PhD thesis, Standford University, 2000.

13 Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

14 Kamal Jain, Mohammad Mahdian, and Amin Saberi. A new greedy approach for facility location problems. In John H. Reif, editor, *STOC*, pages 731–740. ACM, 2002.

15 Kamal Jain and Vijay V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *J. ACM*, 48(2):274–296, 2001.

16 Madhukar R. Korupolu, C. Greg Plaxton, and Rajmohan Rajaraman. Analysis of a local search heuristic for facility location problems. *J. Algorithms*, 37(1):146–188, 2000.

17 Shi Li and Ola Svensson. Approximating k-median via pseudo-approximation. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *STOC*, pages 901–910. ACM, 2013.

18 Jyh-Han Lin and Jeffrey Scott Vitter. epsilon-approximations with minimum packing constraint violation (extended abstract). In S. Rao Kosaraju, Mike Fellows, Avi Wigderson, and John A. Ellis, editors, *STOC*, pages 771–782. ACM, 1992.

Approximation Algorithms for Hypergraph Small Set Expansion and Small Set Vertex Expansion

Anand Louis[*1] **and Yury Makarychev**[†2]

1 **Georgia Tech, Atlanta, USA**
anandl@gatech.edu

2 **Toyota Technological Institute at Chicago, Chicago, USA**
yury@ttic.edu

Abstract

The expansion of a hypergraph, a natural extension of the notion of expansion in graphs, is defined as the minimum over all cuts in the hypergraph of the ratio of the number of the hyperedges cut to the size of the smaller side of the cut. We study the Hypergraph Small Set Expansion problem, which, for a parameter $\delta \in (0, 1/2]$, asks to compute the cut having the least expansion while having at most δ fraction of the vertices on the smaller side of the cut. We present two algorithms. Our first algorithm gives an $\tilde{O}(\delta^{-1}\sqrt{\log n})$ approximation. The second algorithm finds a set with expansion $\tilde{O}(\delta^{-1}(\sqrt{d_{\max} r^{-1} \log r\, \phi^*} + \phi^*))$ in a r–uniform hypergraph with maximum degree $d_{\max}$ (where ϕ^* is the expansion of the optimal solution). Using these results, we also obtain algorithms for the Small Set Vertex Expansion problem: we get an $\tilde{O}(\delta^{-1}\sqrt{\log n})$ approximation algorithm and an algorithm that finds a set with vertex expansion $O\left(\delta^{-1}\sqrt{\phi^V \log d_{\max}} + \delta^{-1}\phi^V\right)$ (where ϕ^V is the vertex expansion of the optimal solution).

For $\delta = 1/2$, Hypergraph Small Set Expansion is equivalent to the hypergraph expansion problem. In this case, our approximation factor of $O(\sqrt{\log n})$ for expansion in hypergraphs matches the corresponding approximation factor for expansion in graphs due to Arora, Rao, and Vazirani.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Approximation Algorithms, Graph Expansion, Hypergraph Expansion, Vertex Expansion

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.339

1 Introduction

The expansion of a hypergraph, a natural extension of the notion of expansion in graphs, is defined as follows.

▶ **Definition 1** (Hypergraph Expansion)**.** Given a hypergraph $H = (V, E)$ on n vertices (each edge $e \in E$ of H is a subset of vertices), we say that an edge $e \in E$ is cut by a set S if $e \cap S \neq \varnothing$ and $e \cap \bar{S} \neq \varnothing$ (i.e. some vertices in e lie in S and some vertices lie outside of S). We denote the set of edges cut by S by $E_{cut}(S)$. The expansion $\phi(S)$ of a set $S \subset V$ ($S \neq \varnothing$, $S \neq V$) in a hypergraph $H = (V, E)$ is defined as $\phi(S) = \frac{|E_{cut}(S)|}{\min(|S|,|\bar{S}|)}$.

* Supported by Santosh Vempala's NSF award CCF-1217793
† Supported by NSF CAREER award CCF-1150062 and NSF award IIS-1302662.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 339–355

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

Hypergraph expansion and related hypergraph partitioning problems are of immense practical importance, having applications in parallel and distributed computing (Catalyurek and Aykanat [5]), VLSI circuit design and computer architecture (Karypis et al. [8]), scientific computing (Devine et al. [7]) and other areas. In spite of this, there has not been much theoretical work on them. In this paper, we study a generalization of the Hypergraph Expansion problem, namely the Hypergraph Small Set Expansion problem.

▶ **Problem 2** (Hypergraph Small Set Expansion Problem)**.** *Given a hypergraph $H = (V, E)$ and a parameter $\delta \in (0, 1/2]$, the Hypergraph Small Set Expansion problem (H-SSE) is to find a set $S \subset V$ of size at most δn that minimizes $\phi(S)$. The value of the optimal solution to H-SSE is called the small set expansion of H. That is, for $\delta \in (0, 1/2]$, the small set expansion $\phi^*_{H,\delta}$ of a hypergraph $H = (V, E)$ is defined as $\phi^*_{H,\delta} = \min_{\substack{S \subset V \\ 0<|S|\leq \delta n}} \phi(S)$.*

Note that for $\delta = 1/2$, the Hypergraph Small Set Expansion Problem is the Hypergraph Expansion Problem.

Small Set Expansion in graphs has attracted a lot of attention recently. The problem was introduced by Raghavendra and Steurer [15], who showed that it is closely related to the Unique Games problem. Raghavendra, Steurer and Tetali [16] designed an algorithm for SSE that finds a set of size $O(\delta n)$ with expansion $O(\sqrt{\phi^* d \log(1/\delta)})$ in d regular graphs (where ϕ^* is the expansion of the optimal solution). Later Bansal, Feige, Krauthgamer, Makarychev, Nagarajan, Naor, and Schwartz gave a $O(\sqrt{\log n \log(1/\delta)})$ approximation algorithm for the problem.

We present analogs of the results of Bansal et al. [4] and Raghavendra, Steurer and Tetali [16] for hypergraphs. Our first result is an $\tilde{O}(\delta^{-1}\sqrt{\log n})$ approximation algorithm[1] for H-SSE (see Theorem 3). Our second result is an algorithm that finds a set with expansion at most $\tilde{O}\left(\delta^{-1}\left(\sqrt{d_{\max}\frac{\log r}{r}\phi^*_{H,\delta}} + \phi^*_{H,\delta}\right)\right)$ if H is an r–uniform hypergraph with maximum degree $d_{\max}$ (see Theorem 4; the result also applies to non-uniform hypergraphs, see Theorem 21).

We note that H-SSE can be reduced to SSE (small set expansion in graphs) if all hyperedges have bounded size. Let r be the size of the largest hyperedge in H. Construct an auxiliary (weighted) graph F on V as follows: pick a vertex in each hyperedge e and connect it in F to all other vertices of e (i.e. replace e with a star); let the weight of an edge f in F be the total weight of the hyperedges $e \in E$ for which f is part of the representative star of e. Then solve SSE in the graph F. It is easy to see that if we solve SSE using an α approximation algorithm, then we get $(r-1)\alpha$ approximation for H-SSE. This approach gives $O(\sqrt{\log n \log(1/\delta)})$ approximation if r is bounded. However, if H is an arbitrary hypergraph, we only get an $O(n\sqrt{\log n \log(1/\delta)})$ approximation. The goal of this paper is to give an approximation guarantee valid for hypergraphs with hyperedges of arbitrary size. We now formally state our main results.

▶ **Theorem 3.** *There is a randomized polynomial-time approximation algorithm for the Hypergraph Small Set Expansion problem that given a hypergraph $H = (V, E)$, and parameters $\varepsilon \in (0, 1)$ and $\delta \in (0, 1/2)$, finds a set $S \subset V$ of size at most $(1 + \varepsilon)\delta n$ such that*

$$\phi(S) \leq O_\varepsilon\left(\delta^{-1} \log \delta^{-1} \log\log \delta^{-1} \cdot \sqrt{\log n} \cdot \phi^*_{H,\delta}\right) = \tilde{O}_\varepsilon\left(\delta^{-1}\sqrt{\log n}\, \phi^*_{H,\delta}\right),$$

(where the constant in the O notation depends polynomially on $1/\varepsilon$). That is, the algorithm gives $O(\sqrt{\log n})$ approximation when δ and ε are fixed.

[1] The $\tilde{O}$–notation hides a $\log \delta^{-1} \log\log \delta^{-1}$ term.

We state our second result, Theorem 4, for r-uniform hypergraphs. We present and prove a more general Theorem 21 that applies to any hypergraphs in Section 6.

▶ **Theorem 4.** *There is a randomized polynomial-time algorithm for the Hypergraph Small Set Expansion problem that given an r–uniform hypergraph $H = (V, E)$ with maximum degree d_{max}, and parameters $\varepsilon \in (0, 1)$ and $\delta \in (0, 1/2)$ finds a set $S \subset V$ of size at most $(1 + \varepsilon)\delta n$ such that*

$$\phi(S) \leq \tilde{O}_\varepsilon \left(\delta^{-1} \left(\sqrt{d_{max} \frac{\log r}{r} \phi^*_{H,\delta}} + \phi^*_{H,\delta} \right) \right).$$

Our algorithms for H-SSE are bi-criteria approximation algorithms in that they output a set S of size at most $(1 + \varepsilon)\delta n$. We note that this is similar to the algorithm of Bansal et al. [4] for SSE, which also finds a set of size at most $(1 + \varepsilon)\delta n$ rather than a set of size at most δn. The algorithm of Raghavendra, Steurer and Tetali [16] finds a set of size $O(\delta n)$. The approximation factor of our first algorithm does not depend on the size of hyperedges in the input hypergraph. It has the same dependence on n as the algorithm of Bansal et al. [4] for SSE. However, the dependence on $1/\delta$ is quasi-linear; whereas it is logarithmic in the algorithm of Bansal et al. [4]. In fact, we show that the integrality gap of the standard SDP relaxation for H-SSE is at least linear in $1/\delta$ (Theorem 22). The approximation guarantee of our second algorithm is analogous to that of the algorithm of Raghavendra, Steurer and Tetali [16].

Small Set Vertex Expansion

Our techniques can also be used to obtain an approximation algorithm for Small Set Vertex Expansion (SSVE) in graphs.

▶ **Problem 5** (Small Set Vertex Expansion Problem)**.** *Given graph $G = (V, E)$, the vertex expansion of a set $S \subset V$ is defined as*

$$\phi^V(S) = \frac{|\{u \in \bar{S} : \exists\, v \in S \text{ such that } \{u, v\} \in E\}|}{|S|}$$

Given a parameter $\delta \in (0, 1/2]$, the Small Set Vertex Expansion problem (SSVE) is to find a set $S \subset V$ of size at most δn that minimizes $\phi^V(S)$. The value of the optimal solution to SSVE is called the small set vertex expansion of G. That is, for $\delta \in (0, 1/2]$, the small set expansion $\phi^V_{G,\delta}$ of a graph $G = (V, E)$ is defined as

$$\phi^V_{G,\delta} = \min_{\substack{S \subset V \\ 0 < |S| \leq \delta n}} \phi^V(S).$$

Small Set Vertex Expansion recently gained interest due to its connection to obtaining sub-exponential-time, constant factor approximation algorithms for many combinatorial problems like Sparsest Cut and Graph Coloring [1, 12]. Using a reduction from vertex expansion in graphs to hypergraph expansion, we can get an approximation algorithm for SSVE having the same approximation guarantee as that for H-SSE.

▶ **Theorem 6.** *There exist absolute constants $c_1, c_2 \in \mathbb{R}^+$ such that for every graph $G = (V, E)$, there exists a polynomial time computable hypergraph $H = (V', E')$ such that $c_1\phi^*_{H,\delta} \leq \phi^V_{G,\delta} \leq c_2\phi^*_{H,\delta}$. Also, $\eta^H_{max} \leq \log_2(d_{max} + 1)$, where d_{max} is the maximum degree of G (where η^H_{max} is defined in Definition 20).*

From this theorem, Theorem 3 and Theorem 21 we immediately get algorithms for SSVE.

▶ **Theorem 7** (Corollary to Theorem 3 and Theorem 6). *There is a randomized polynomial-time approximation algorithm for the Small Set Vertex Expansion problem that given a graph $G = (V, E)$, and parameters $\varepsilon \in (0, 1)$ and $\delta \in (0, 1/2))$ finds a set $S \subset V$ of size at most $(1 + \varepsilon)\delta n$ such that*

$$\phi^V(S) \leq O_\varepsilon\left(\sqrt{\log n}\,\delta^{-1}\log\delta^{-1}\log\log\delta^{-1}\cdot\phi^V_{G,\delta}\right),$$

That is, the algorithm gives $O(\sqrt{\log n})$ approximation when δ and ε are fixed.

▶ **Theorem 8** (Corollary to Theorem 21 and Theorem 6). *There is a randomized polynomial-time algorithm for the Small Set Vertex Expansion problem that given a graph $G = (V, E)$ of maximum degree d_{max}, parameters $\varepsilon \in (0, 1)$ and $\delta \in (0, 1/2)$ finds a set $S \subset V$ of size at most $(1 + \varepsilon)\delta n$ such that*

$$\begin{aligned}\phi^V(S) &\leq O_\varepsilon\left(\sqrt{\phi^V_{G,\delta}\log d_{max}}\cdot\delta^{-1}\log\delta^{-1}\log\log\delta^{-1} + \delta^{-1}\phi^V_{G,\delta}\right)\\ &= \tilde{O}_\varepsilon\left(\delta^{-1}\sqrt{\phi^V_{G,\delta}\log d_{max}} + \delta^{-1}\phi^V_{G,\delta}\right).\end{aligned}$$

We note that the Small Set Vertex Expansion problem for $\delta = 1/2$ is just the Vertex Expansion problem. In that case, Theorem 8 gives the same approximation guarantee as the algorithm of Louis, Raghavendra and Vempala [13].

Techniques. Our general approach to solving H-SSE is similar to the approach of Bansal et al. [4]. We recall how the algorithm of Bansal et al. [4] for (graph) SSE works. The algorithm solves a semidefinite programming relaxation for SSE and gets an SDP solution. The SDP solution assigns a vector $\bar{u}$ to each vertex u. Then the algorithm generates *an orthogonal separator*. Informally, an orthogonal separator S with distortion D is a random subset of vertices such that

- **(a)** If $\bar{u}$ and $\bar{v}$ are close to each other then the probability that u and v are separated by S is small; namely, it is at most $\alpha D\|\bar{u} - \bar{v}\|^2$, where α is a normalization factor such that $\Pr(u \in S) = \alpha\|\bar{u}\|^2$.
- **(b)** If the angle between $\bar{u}$ and $\bar{v}$ is larger than a certain threshold, then the probability that both u and v are in S is much smaller than the probability that one of them is in S.

Bansal et al. [4] showed that condition (b) together with SDP constraints implies that S is of size at most $(1 + \varepsilon)\delta n$ with sufficiently high probability. Then condition (a) implies that the expected number of cut edges is at most D times the SDP value. That means that S is a D–approximate solution to SSE.

If we run this algorithm on an instance of H-SSE, we will still find a set of size at most $(1 + \varepsilon)\delta n$, but the cost of the solution might be very high. Indeed, consider a hyperedge e. Even though every two vertices u and v in e are *unlikely* to be separated by S, at least one pair out of $\binom{|e|}{2}$ pairs of vertices is quite likely to be separated by S; hence, e is quite likely to be cut by S. To deal with this problem, we develop *hypergraph orthogonal separators*. In the definition of a hypergraph orthogonal separator, we strengthen condition (a) by requiring that a hyperedge e is cut by S with small probability if all vertices in e are close to each other. Specifically, we require that

$$\Pr(e \text{ is cut by } S) \leq \alpha D \max_{u,v\in e}\|\bar{u} - \bar{v}\|^2. \tag{1}$$

We show that there is a hypergraph orthogonal separator with distortion proportional to $\sqrt{\log n}$ (the distortion also depends on parameters of the orthogonal separator). Plugging this hypergraph orthogonal separator in the algortihm of Bansal et al. [4], we get Theorem 3. We also develop another variant of hypergraph orthogonal separators, ℓ_2–ℓ_2^2 orthogonal separators. An ℓ_2–ℓ_2^2 orthogonal separator with ℓ_2–distortion $D_{\ell_2}(r)$ and ℓ_2^2–distortion $D_{\ell_2^2}$ satisfies the following condition[2]

$$\Pr(e \text{ is cut by } S) \le \alpha D_{\ell_2}(|e|) \cdot \min_{w\in E} \|\bar{w}\| \cdot \max_{u,v\in e} \|\bar{u}-\bar{v}\| + \alpha D_{\ell_2^2} \cdot \max_{u,v\in e} \|\bar{u}-\bar{v}\|^2. \tag{2}$$

We show that there is an ℓ_2-ℓ_2^2 hypergraph orthogonal separator whose ℓ_2 and ℓ_2^2 distortions do not depend on n (in contrast, there is no hypergraph orthogonal separator whose distortion does not depend on n). This result yields Theorem 4.

We now give a brief conceptual overview of our construction of hypergraph orthogonal separators. We use the framework developed by Chlamtac, Makarychev, and Makarychev in [6, Section 4.3] for (graph) orthogonal separators. For simplicity, we ignore vector normalization steps in this overview; we do not explain how we take into account vector lengths. Note, however, that these normalization steps are crucial. We first design a procedure that partitions the hypergraph into two pieces (the procedure labels every vertex with either 0 or 1). In a sense, each set S in the partition is a "very weak" hypergraph orthogonal separator. It satisfies property (1) with $D_0 \sim \sqrt{\log n}\log\log(1/\delta)$ and $\alpha_0 = 1/2$ and a weak variant of property (b): if the angle between vectors $\bar{u}$ and $\bar{v}$ is larger than the threshold then events $u \in S$ and $v \in S$ are "almost" independent. We repeat the procedure $l = \log_2(1/\delta) + O(1)$ times and obtain a partition of graph into $2^l = O(1/\delta)$ pieces. Then we randomly choose one set S among them; this set S is our hypergraph orthogonal separator. Note that by running the procedure many times we decrease exponentially in l the probability that two vertices, as in condition (b), belong to S. So condition (b) holds for S. Also, we affect the distortion in (1) in two ways. First, the probability that the edge is cut increases by a factor of l. That is, we get $\Pr(e \text{ is cut by } S) \le l \times \alpha_0 D_0 \max_{u,v\in e} \|\bar{u}-\bar{v}\|^2$. Second, the probability that we choose a vertex u goes down from $\|\bar{u}\|^2/2$ to $\Omega(\delta)\|\bar{u}\|^2$ since roughly speaking we choose one set S among $O(1/\delta)$ possible sets. That is, the parameter α of S is $\Omega(\delta)$. Therefore, $\Pr(e \text{ is cut by } S) \le \alpha(\alpha_0 l D_0/\alpha)\max_{u,v\in e} \|\bar{u}-\bar{v}\|^2$. That is, we get a hypergraph orthogonal separator with distortion $(\alpha_0 l D_0/\alpha) \sim \tilde{O}(\delta^{-1}\sqrt{\log n})$. The construction of ℓ_2–ℓ_2^2 orthogonal separators is similar but a bit more technical.

Organization. We present our SDP relaxation and introduce our main technique, hypergraph orthogonal separators, in Section 2. We describe our first algorithm for H-SSE in Section 3, and then describe an algorithm that generates hypergraph orthogonal separators in Section 4. We define ℓ_2–ℓ_2^2 hypergraph orthogonal separators, give an algorithm that generates them, and then present our second algorithm for H-SSE in Section 5 and Section 6. Finally, we show a simple SDP integrality gap for H-SSE in Section 7. This integrality gap also gives a lower bound on the quality of m-orthogonal separators. We give a proof of Theorem 6 in Section 8.

[2] It may look strange that we have two terms in the bound. One may expect that we can either have only term $D_{\ell_2^2}\max_{u,v\in e}\|\bar{u}-\bar{v}\|^2$ (as in the previous definition) or only term $D_{\ell_2}(|e|)\cdot\min_{w\in E}\|\bar{w}\|\cdot\max_{u,v\in e}\|\bar{u}-\bar{v}\|$. However, the latter is not possible — there is no ℓ_2–ℓ_2^2 separator with $D_{\ell_2^2}=0$.

$$\text{minimize} \sum_{e\in E} \max_{u,v\in e} \|\bar{u}-\bar{v}\|^2 \tag{3}$$

subject to:

$$\sum_{v\in V} \langle \bar{u},\bar{v}\rangle \le \delta n \cdot \|\bar{u}\|^2 \qquad \text{for every } u\in V \tag{4}$$

$$\sum_{u\in V} \|\bar{u}\|^2 = 1 \tag{5}$$

$$\|\bar{u}-\bar{v}\|^2 + \|\bar{v}-\bar{w}\|^2 \ge \|\bar{u}-\bar{w}\|^2 \qquad \text{for every } u,v,w\in V \tag{6}$$

$$0 \le \langle \bar{u},\bar{v}\rangle \le \|u\|^2 \qquad \text{for every } u,v\in V. \tag{7}$$

Figure 1 SDP relaxation for H-SSE.

2 Preliminaries

2.1 SDP Relaxation for Hypergraph Small Set Expansion

We use the SDP relaxation for H-SSE shown in Figure 1. There is an SDP variable $\bar{u}$ for every vertex $u \in V$.

Every combinatorial solution S (with $|S| \le \delta n$) defines the corresponding (intended) SDP solution: $\bar{u} = \frac{e}{\sqrt{|S|}}$, if $u \in S$; $\bar{u} = 0$, otherwise, where e is a fixed unit vector. It is easy to see that this solution satisfies all SDP constraints. Note that $\max_{u,v\in e} \|\bar{u}-\bar{v}\|^2$ is equal to $1/|S|$, if e is cut, and to 0, otherwise. Therefore, the objective function equals

$$\sum_{e\in E} \max_{u,v\in e} \|\bar{u}-\bar{v}\|^2 = \sum_{e\in E_{cut}(S)} \frac{1}{|S|} = \frac{E_{cut}(S)}{S} = \phi(S).$$

Thus our SDP for H-SSE is indeed a relaxation.

2.2 Hypergraph Orthogonal Separators

The main technical tool for proving Theorem 3 is *hypergraph orthogonal separators*. *Orthogonal separators* were introduced by Chlamtac, Makarychev, and Makarychev [6] (see also Bansal et al. [4], Louis and Makarychev [9], and Makarychev and Makarychev [14]) and were previously used for solving Unique Games and various graph partitioning problems. In this paper, we extend the technique of orthogonal separators to hypergraphs and introduce hypergraph orthogonal separators. We then use hypergraph orthogonal separators to solve H-SSE. In Section 5, we introduce another version of hypergraph orthogonal separators, ℓ_2–ℓ_2^2 hypergraph orthogonal separators, and then use them to prove Theorem 4 and Theorem 21.

▶ **Definition 9** (Hypergraph Orthogonal Separators). Let $\{\bar{u} : u \in V\}$ be a set of vectors in the unit ball that satisfy ℓ_2^2–triangle inequalities (6) and (7). We say that a random set $S \subset V$ is *a hypergraph m-orthogonal separator* with distortion $D \ge 1$, probability scale $\alpha > 0$, and separation threshold $\beta \in (0,1)$ if it satisfies the following properties.

1. For every $u \in V$, $\Pr(u \in S) = \alpha\|\bar{u}\|^2$.
2. For every u and v such that $\|\bar{u}-\bar{v}\|^2 \ge \beta \min(\|\bar{u}\|^2, \|\bar{v}\|^2)$

$$\Pr\left(u \in S \text{ and } v \in S\right) \le \alpha \frac{\min(\|\bar{u}\|^2, \|\bar{v}\|^2)}{m}.$$

3. For every $e \subset V$, $\Pr\left(e \text{ is cut by } S\right) \le \alpha D \max_{u,v\in e} \|\bar{u}-\bar{v}\|^2$.

The definition of a hypergraph m-orthogonal separator is similar to that of a (graph) m-orthogonal separator: a random set S is an m-orthogonal separator if it satisfies properties 1, 2, and property 3′, which is property 3 restricted to edges e of size 2.

3′. For every (u,v), $\Pr(e \text{ is cut by } S) \le \alpha D\|\bar{u}-\bar{v}\|^2$.

In this paper, we design an algorithm that generates a hypergraph m-orthogonal separator with distortion $O_\beta(\sqrt{\log n}\cdot m\log m\log\log m)$. We note that the distortion of *any* hypergraph orthogonal separator must depend on m at least linearly (see Section 7). We remark that there are two constructions of (graph) orthogonal separators, "orthogonal separators via ℓ_1" and "orthogonal separators via ℓ_2", with distortions, $O_\beta(\sqrt{\log n}\log m)$ and $O_\beta(\sqrt{\log n\log m})$, respectively (presented in [6]). Our construction of hypergraph orthogonal separators uses the framework of orthogonal separators via ℓ_1. We prove the following theorem in Section 4.

▶ **Theorem 10.** *There is a polynomial-time randomized algorithm that given a set of vertices V, a set of vectors $\{\bar{u}\}$ satisfying ℓ_2^2–triangle inequalities (6) and (7), parameters $m \ge 2$ and $\beta \in (0,1)$, generates a hypergraph m-orthogonal separator with probability scale $\alpha \ge 1/n$ and distortion $D = O(\beta^{-1} m\log m\log\log m \times \sqrt{\log n})$.*

3 Algorithm for Hypergraph Small Set Expansion

In this section, we present our algorithm for Hypergraph Small Set Expansion. Our algorithm uses hypergraph orthogonal separators that we describe in Section 4. We use the approach of Bansal et al. [4]. Suppose that we are given a polynomial-time algorithm that generates hypergraph m-orthogonal separators with distortion $D(m,\beta)$ (with probability scale $\alpha > 1/\operatorname{poly}(n)$). We show how to get a $D^* = 4D(4/(\varepsilon\delta),\varepsilon/4)$ approximation for H-SSE.

▶ **Theorem 11.** *There is a randomized polynomial-time approximation algorithm for the Hypergraph Small Set Expansion problem that given a hypergraph $H=(V,E)$, and parameters $\varepsilon \in (0,1)$ and $\delta \in (0,1/2)$ finds a set $S \subset V$ of size at most $(1+\varepsilon)\delta n$ such that $\phi(S) \le 4D(4/(\varepsilon\delta),\varepsilon/4)\cdot\phi^*_{H,\delta}$.*

Proof. We solve the SDP relaxation for H-SSE and obtain an SDP solution $\{\bar{u}\}$. Denote the SDP value by sdp-cost. Consider a hypergraph orthogonal separator S with $m = 4/(\varepsilon\delta)$ and $\beta = \varepsilon/4$. Define a set S':

$$S' = \begin{cases} S, & \text{if } |S| \le (1+\varepsilon)\delta n,\\ \varnothing, & \text{otherwise.}\end{cases}$$

Clearly, $|S'| \le (1+\varepsilon)\delta n$. Bansal et al. [4] showed that $\Pr(u\in S') \in \left[\frac{\alpha}{2}\|\bar{u}\|^2, \alpha\|\bar{u}\|^2\right]$ for every $u \in V$ (see also Theorem A.1 in [14]). Note that

$$\Pr(S' \text{ cuts edge } e) \le \Pr(S \text{ cuts edge } e) \le \alpha D^* \max_{u,v\in e}\|\bar{u}-\bar{v}\|^2.$$

where D^* denotes $D(4/(\varepsilon\delta),\varepsilon/4)$ for the sake of brevity. Let $Z = |S'| - \frac{|E_{cut}(S')|}{4D^*\cdot\text{sdp-cost}}$. We have,

$$\mathbb{E}[Z] = \mathbb{E}[|S'|] - \frac{\mathbb{E}[|E_{cut}(S')|]}{4D^*\cdot\text{sdp-cost}} \ge \sum_{u\in V}\frac{\alpha}{2}\cdot\|\bar{u}\|^2 - \frac{\sum_{e\in E}\alpha D^*\max_{u,v\in e}\|\bar{u}-\bar{v}\|^2}{4D^*\cdot\text{sdp-cost}}$$
$$= \frac{\alpha}{2} - \frac{1}{4D^*\cdot\text{sdp-cost}}\times\alpha D^*\,\text{sdp-cost} = \frac{\alpha}{4}.$$

Since $Z \le |S'| \le (1+\varepsilon)\delta n < n$ (always), by Markov's inequality, we have $\Pr(Z > 0) \ge \alpha/(4n)$ and hence $\Pr(|E_{cut}(S')|/|S'| < 4D^* \cdot \text{sdp-cost}) \ge \alpha/(4n)$.

We sample S independently $4n/\alpha$ times and return the first set S' such that $\frac{|E_{cut}(S')|}{|S'|} < 4D^* \cdot \text{sdp-cost}$. This gives a set S' such that $|S'| \le (1+\varepsilon)\delta n$, and $\phi(S') \le 4D^*\phi^*_{H,\delta}$. The algorithm succeeds (finds such a set S') with a constant probability. By repeating the algorithm n times, we can make the success probability exponentially close to 1. ◀

In Section 4, we describe how to generate an m-hypergraph orthogonal separator with distortion $D = O\big(\sqrt{\log n} \times \beta^{-1} m \log m \log\log m\big)$. That gives us an algorithm for H-SSE with approximation factor $O_\varepsilon\left(\delta^{-1}\log\delta^{-1}\log\log\delta^{-1} \times \sqrt{\log n}\right)$.

4 Generating Hypergraph Orthogonal Separators

In this section, we present an algorithm that generates a hypergraph m-orthogonal separator. At the high level, the algorithm is similar to the algorithm for generating orthogonal separators from Section 4.3 in [6]. We use a different procedure for generating words $W(u)$ (see below) and set parameters differently; also the analysis of our algorithm is different.

In our algorithm, we use a "normalization" map φ from [6]. Map φ maps a set $\{\bar{u}\}$ of vectors satisfying ℓ_2^2–triangle inequalities (6) and (7) to $\mathbb{R}^n$. It has the following properties.

1. For all vertices u, v, w, $\|\varphi(\bar{u}) - \varphi(\bar{v})\|_2^2 + \|\varphi(\bar{v}) - \varphi(\bar{w})\|_2^2 \ge \|\varphi(\bar{u}) - \varphi(\bar{w})\|_2^2$.
2. For all nonzero vertices u and v, $\langle\varphi(\bar{u}), \varphi(\bar{v})\rangle = \frac{\langle\bar{u},\bar{v}\rangle}{\max(\|\bar{u}\|^2, \|\bar{v}\|^2)}$.
3. In particular, for every $\bar{u} \ne 0$, $\|\varphi(\bar{u})\|_2^2 = \langle\varphi(\bar{u}), \varphi(\bar{u})\rangle = 1$. Also, $\varphi(0) = 0$.
4. For all non-zero vectors $\bar{u}$ and $\bar{v}$, $\|\varphi(\bar{u}) - \varphi(\bar{v})\|_2^2 \le \frac{2\,\|\bar{u}-\bar{v}\|^2}{\max(\|\bar{u}\|^2, \|\bar{v}\|^2)}$.

We also use the following theorem of Arora, Lee, and Naor [2] (see also [3]).

▶ **Theorem 12** (Arora, Lee, and Naor (2005), Theorem 3.1). *There exist constants $C \ge 1$ and $p \in (0, 1/4)$ such that for every n unit vectors x_u ($u \in V$), satisfying ℓ_2^2–triangle inequalities (6), and every $\Delta > 0$, the following holds. There exists a random subset U of V such that for every $u, v \in V$ with $\|x_u - x_v\|^2 \ge \Delta$, $\Pr\left(u \in U \text{ and } d(v,U) \ge \frac{\Delta}{C\sqrt{\log n}}\right) \ge p$, where $d(v,U) = \min_{u\in U}\|x_u - x_v\|^2$.*

First we describe an algorithm that randomly assigns each vertex u a symbol, either 0 or 1. Then we use this algorithm to generate an orthogonal separator.

▶ **Lemma 13.** *There is a randomized polynomial-time algorithm that given a finite set V, unit vectors $\varphi(\bar{u})$ for $u \in V$ satisfying ℓ_2^2-triangle inequalities and a parameter $\beta \in (0,1)$, returns a random assignment $\omega : V \to \{0,1\}$ that satisfies the following properties.*

- *For every u and v such that $\|\varphi(\bar{u}) - \varphi(\bar{v})\|^2 \ge \beta$, we have $\Pr(\omega(u) \ne \omega(v)) \ge 2p$, where $p > 0$ is the constant from Theorem 12.*
- *For every set $e \subset V$ of size at least 2,*

$$\Pr(\omega(u) \ne \omega(v) \text{ for some } u, v \in e) \le O(\beta^{-1}\sqrt{\log n}\max_{u,v\in e}\|\varphi(\bar{u}) - \varphi(\bar{v})\|^2).$$

Proof. Let U be the random set from Theorem 12 for vectors $x_u = \varphi(\bar{u})$ and $\Delta = \beta$. Choose $t \in (0, 1/(C\sqrt{\log n}))$ uniformly at random. Let

$$\omega(u) = \begin{cases} 0, & \text{if } d(U_i, u) \le t, \\ 1, & \text{otherwise.} \end{cases}$$

Consider first vertices u and v such that $\|\varphi(\bar{u}) - \varphi(\bar{v})\|^2 \geq \beta$. By Theorem 12,

$$\Pr\left(u \in U \text{ and } d(v,U) \geq \frac{\Delta}{C\sqrt{\log n}}\right) \geq p \quad \text{and} \quad \Pr\left(v \in U \text{ and } d(u,U) \geq \frac{\Delta}{C\sqrt{\log n}}\right) \geq p.$$

Note that in the former case, when $u \in U$ and $d(v,U) \geq \frac{\Delta}{C\sqrt{\log n}}$, we have $\omega(u) = 0$ and $\omega(v) = 1$; in the latter case, when $v \in U$ and $d(u,U) \geq \frac{\Delta}{C\sqrt{\log n}}$, we have $\omega(v) = 0$ and $\omega(u) = 1$. Therefore, the probability that $\omega(u) \neq \omega(v)$ is at least $2p$.

Now consider a set $e \subset V$ of size at least 2. Let $\tau_m = \min_{w\in e} d(U, \varphi(\bar{w}))$ and $\tau_M = \max_{w\in e} d(U, \varphi(\bar{w}))$. We have, $\tau_M - \tau_m \leq \max_{u,v\in e} \|\varphi(\bar{u}) - \varphi(\bar{v})\|^2$. Note that if $t < \tau_m$ then $\omega(u) = 1$ for all $u \in e$; if $t \geq \tau_M$ then $\omega(u) = 0$ for all $u \in e$. Thus $\omega(u) \neq \omega(v)$ for some $u, v \in e$ only if $t \in [\tau_m, \tau_M)$. Since the probability density of the random variable t is at most $C\sqrt{\log n}$, we get,

$$\Pr\left(\exists\, u, v \in e : \omega(u) \neq \omega(v)\right) \leq \Pr\left(t \in [\tau_m, \tau_M)\right) \leq \frac{C\sqrt{\log n}}{\beta} \max_{u,v\in e} \|\bar{u} - \bar{v}\|^2.$$

◀

We now amplify the result of Lemma 13.

▶ **Lemma 14.** *There is a randomized polynomial time algorithm that given V, vectors $\varphi(\bar{u})$ and $\beta \in (0,1)$ as in Lemma 13, and a parameter $m \geq 2$, returns a random assignment $\tilde{\omega} : V \to \{0,1\}$ such that:*

- *For every u and v such that $\|\varphi(\bar{u}) - \varphi(\bar{v})\|^2 \geq \beta$, $\Pr\left(\tilde{\omega}(u) \neq \tilde{\omega}(v)\right) \geq 1/2 - 1/\log_2 m$.*
- *For every set $e \subset V$ of size at least 2,*

$$\Pr\left(\tilde{\omega}(u) \neq \tilde{\omega}(v) \text{ for some } u, v \in e\right) \leq O(\beta^{-1}\sqrt{\log n} \cdot \log\log m \cdot \max_{u,v\in e} \|\varphi(\bar{u}) - \varphi(\bar{v})\|^2).$$

We independently sample $K = \max\left(\left\lceil \frac{\log_2 \log_2 m}{-\log_2(1-4p)} \right\rceil, 1\right)$ assignments $\omega_1, \ldots, \omega_K$, and let $\tilde{\omega}(u) = \omega_1(u) \oplus \cdots \oplus \omega_K(u)$ (where $\oplus$ denotes addition modulo 2). It is easy to see that the assignment $\tilde{\omega}$ satisfies the required properties. We defer the proof to the full version of the paper [10, Section E].

We are now ready to present our algorithm.

1. Set $l = \lceil \log_2 m/(1 - \log_2(1 + 2/\log_2 m))\rceil = \log_2 m + O(1)$.
2. Sample l independent assignments $\tilde{\omega}_1, \ldots, \tilde{\omega}_l$ using Lemma 14.
3. For every vertex u, define word $W(u) = \tilde{\omega}_1(u) \ldots \tilde{\omega}_l(u) \in \{0,1\}^l$.
4. If $n \geq 2^l$, pick a word $W \in \{0,1\}^l$ uniformly at random. If $n < 2^l$, pick a random word $W \in \{0,1\}^l$ so that $\Pr_W(W = W(u)) = 1/n$ for every $u \in V$. This is possible since the number of distinct words constructed in step 3 is at most n (we may pick a word W not equal to any $W(u)$).
5. Pick $r \in (0,1)$ uniformly at random.
6. Let $S = \left\{u \in V : \|\bar{u}\|^2 \geq r \text{ and } W(u) = W\right\}$.

▶ **Theorem 15.** *Random set S is a hypergraph m-orthogonal separator with distortion $D = O\big(\sqrt{\log n} \times \frac{m \log m \log\log m}{\beta}\big)$, probability scale $\alpha \geq 1/n$ and separation threshold β.*

Proof. We verify that S satisfies properties 1–3 in the definition of a hypergraph m-orthogonal separator with $\alpha = \max(1/2^l, 1/n)$.

Property 1. We compute the probability that $u \in S$. Observe that $u \in S$ if and only if $W(u) = W$ and $r \le \|\bar{u}\|^2$ (these two events are independent). If $n \ge 2^l$, the probability that $W = W(u)$ is $1/2^l$ since we choose W uniformly at random from $\{0,1\}^l$; if $n < 2^l$ the probability is $1/n$. That is, $\Pr(W = W(u)) = \max(1/2^l, 1/n) = \alpha$. The probability that $r \le \|\bar{u}\|^2$ is $\|\bar{u}\|^2$. We conclude that property 1 holds.

Property 2. Consider two vertices u and v such that $\|\bar{u}-\bar{v}\|^2 \ge \beta \min(\|\bar{u}\|^2, \|\bar{v}\|^2)$. Assume without loss of generality that $\|\bar{u}\|^2 \le \|\bar{v}\|^2$. Note that $u, v \in S$ if and only if $r \le \|\bar{u}\|^2$ and $W = W(u) = W(v)$. We first upper bound the probability that $W(u) = W(v)$. We have, $2\langle \bar{u}, \bar{v}\rangle = \|\bar{u}\|^2 + \|\bar{v}\|^2 - \|\bar{u} - \bar{v}\|^2 \le (1-\beta)\|\bar{u}\|^2 + \|\bar{v}\|^2 \le (2-\beta)\|\bar{v}\|^2$. Therefore, $2\langle \bar{u}, \bar{v}\rangle / \|\bar{v}\|^2 \le 2-\beta$. Hence, $\|\varphi(\bar{u}) - \varphi(\bar{v})\|^2 = 2 - 2\langle \varphi(\bar{u}), \varphi(\bar{v})\rangle = 2 - \frac{2\langle \bar{u}, \bar{v}\rangle}{\max(\|\bar{u}\|^2, \|\bar{v}\|^2)} \ge \beta = \Delta$. From Lemma 14 we get that $\Pr(\tilde{\omega}_i(u) \ne \tilde{\omega}_i(v)) \ge \frac{1}{2} - \frac{1}{\log_2 m}$ for every i. The probability that $W(u) = W(v)$ is at most $(\frac{1}{2} + \frac{1}{\log_2 m})^l \le 1/m$. Therefore we have as required,

$$\begin{aligned}\Pr(u \in S, v \in S) = \Pr\left(r \le \min(\|\bar{u}\|^2, \|\bar{v}\|^2)\right) \times \Pr(W = W(u) = W(v) \mid W(u) = W(v)) \\ \times \Pr(W(u) = W(v)) \le \min(\|\bar{u}\|^2, \|\bar{v}\|^2) \times \alpha \times (1/m).\end{aligned}$$

Property 3. Let e be an arbitrary subset of V, $|e| \ge 2$. Let $\rho_m = \min_{w \in e} \|\bar{w}\|^2$ and $\rho_M = \max_{w \in e} \|\bar{w}\|^2$. Note that $\rho_M - \rho_m = \|\bar{w}_1\|^2 - \|\bar{w}_2\|^2 \le \|\bar{w}_1 - \bar{w}_2\|^2 \le \max_{u,v \in e} \|\bar{u} - \bar{v}\|^2$, for some $w_1, w_2 \in e$. Here we used that SDP constraint (7) implies that $\|\bar{w}_1\|^2 - \|\bar{w}_2\|^2 \le \|\bar{w}_1 - \bar{w}_2\|^2$.

Let $A = \{u \in e : \|\bar{u}\|^2 \ge r\}$. Note that $S \cap e = \{u \in A : W(u) = W\}$. Therefore, if e is cut by S then one of the following events happens.

- Event $\mathcal{E}_1$: $A \ne e$ and $S \cap e \ne \varnothing$.
- Event $\mathcal{E}_2$: $A = e$ and $A \cap S \ne \varnothing$, $A \cap S \ne A$.

If $\mathcal{E}_1$ happens then $r \in [\rho_m, \rho_M]$ since $A \ne e$ and $A \ne \varnothing$. We have,

$$\Pr(\mathcal{E}_1) \le \Pr(r \in (\rho_m, \rho_M]) \le |\rho_M - \rho_m| \le \max_{u,v \in e} \|\bar{u} - \bar{v}\|^2.$$

If $\mathcal{E}_2$ happens then (1) $r \le \rho_m$ (since $A = e$) and (2) $W(u) \ne W(v)$ for some $u, v \in e$. The probability that $r \le \rho_m$ is ρ_m. We now upper bound the probability that $W(u) \ne W(v)$ for some $u, v \in e$. For each $i \in \{1, \ldots, l\}$,

$$\begin{aligned}\Pr(\tilde{\omega}_i(u) \ne \tilde{\omega}_i(v) \text{ for some } u, v \in e) &\le O(\beta^{-1}\sqrt{\log n} \cdot \log\log m) \max_{u,v \in e} \|\varphi(\bar{u}) - \varphi(\bar{v})\|^2 \\ &\le O(\beta^{-1}\sqrt{\log n} \cdot \log\log m) \max_{u,v \in e} \frac{2\|\bar{u} - \bar{v}\|^2}{\min(\|\bar{u}\|^2, \|\bar{v}\|^2)} \\ &\le O(\beta^{-1}\sqrt{\log n} \cdot \log\log m) \times \rho_m^{-1} \times \max_{u,v \in e} \|\bar{u} - \bar{v}\|^2.\end{aligned}$$

By the union bound over $i \in \{1, \ldots, l\}$, the probability that $W(u) \ne W(v)$ for some $u, v \in e$ is at most $O(l \times \beta^{-1}\sqrt{\log n} \cdot \log\log m) \times \rho_m^{-1} \times \max_{u,v \in e} \|\bar{u} - \bar{v}\|^2$. Therefore,

$$\begin{aligned}\Pr(\mathcal{E}_2) &\le \rho_m \times O(l \times \beta^{-1}\sqrt{\log n}\, \log\log m) \times \rho_m^{-1} \times \max_{u,v \in e} \|\bar{u} - \bar{v}\|^2 \\ &\le O(\beta^{-1}\sqrt{\log n}\, \log m \log\log m) \times \max_{u,v \in e} \|\bar{u} - \bar{v}\|^2.\end{aligned}$$

We get that the probability that e is cut by S is at most

$$\Pr(\mathcal{E}_1) + \Pr(\mathcal{E}_2) \le O(\beta^{-1}\sqrt{\log n}\, \log m \log\log m) \times \max_{u,v \in e} \|\bar{u} - \bar{v}\|^2.$$

For $D = O(\beta^{-1}\sqrt{\log n}\, \log m \log\log m)/\alpha$ we get $\Pr(e \text{ is cut by } S) \le \alpha D \max_{u,v \in e} \|\bar{u} - \bar{v}\|^2$. Note that $\alpha \ge 1/2^l \ge \Omega(1/m)$. Thus $D \le O(\beta^{-1}\sqrt{\log n}\, m \log m \log\log m)$. ◀

5 ℓ_2–ℓ_2^2 Hypergraph Orthogonal Separators

In this section, we present another variant of hypergraph orthogonal separators, which we call ℓ_2–ℓ_2^2 hypergraph orthogonal separators. The advantage of ℓ_2–ℓ_2^2 hypergraph orthogonal separators is that their distortions do not depend on n (the number of vertices). Then in Section 6, we use ℓ_2–ℓ_2^2 hypergraph orthogonal separators to prove Theorem 21 (which, in turn, implies Theorem 4).

▶ **Definition 16** (ℓ_2–ℓ_2^2 Hypergraph Orthogonal Separator)**.** Let $\{\bar{u} : u \in V\}$ be a set of vectors in the unit ball. We say that a random set $S \subset V$ is *a ℓ_2–ℓ_2^2 hypergraph m-orthogonal separator* with ℓ_2–distortion $D_{\ell_2} : \mathbb{N} \to \mathbb{R}$, ℓ_2^2–distortion $D_{\ell_2^2}$, probability scale $\alpha > 0$, and separation threshold $\beta \in (0,1)$ if it satisfies the following properties.

1. For every $u \in V$, $\Pr(u \in S) = \alpha\|\bar{u}\|^2$.
2. For every u and v such that $\|\bar{u} - \bar{v}\|^2 \geq \beta \min(\|\bar{u}\|^2, \|\bar{v}\|^2)$

$$\Pr(u \in S \text{ and } v \in S) \leq \alpha \frac{\min(\|\bar{u}\|^2, \|\bar{v}\|^2)}{m}.$$

3. For every $e \subset V$,

$$\Pr(e \text{ is cut by } S) \leq \alpha D_{\ell_2^2} \cdot \max_{u,v\in e} \|\bar{u} - \bar{v}\|^2 + \alpha D_{\ell_2}(|e|) \cdot \min_{w\in e} \|barw\| \cdot \max_{u,v\in e} \|\bar{u} - \bar{v}\|.$$

(This definition differs from Definition 9 only in item 3.)

▶ **Theorem 17.** *There is a polynomial-time randomized algorithm that given a set of vertices V, a set of vectors $\{\bar{u}\}$ satisfying ℓ_2^2–triangle inequalities, and parameters m and β generates an ℓ_2–ℓ_2^2 hypergraph m-orthogonal separator with probability scale $\alpha \geq 1/n$ and distortions:*

$$D_{\ell_2^2} = O(m) \qquad \text{and} \qquad D_{\ell_2}(r) = O(\beta^{-1/2}\sqrt{\log r}\, m \log m \log\log m).$$

Note that distortions $D_{\ell_2^2}$ and D_{ℓ_2} do not depend on n.

The algorithm and its analysis are very similar to those in the proof of Theorem 10. The only difference is that we use another procedure to generate random assignments $\omega : V \to \{0,1\}$. The following lemma is an analog of Lemma 13.

▶ **Lemma 18.** *There is a randomized polynomial time algorithm that given a finite set V, vectors $\varphi(\bar{u})$ for $u \in V$, satisfying ℓ_2^2 triangle inequalities, and a parameter $\beta \in (0,1)$, returns a random assignment $\omega : V \to \{0,1\}$ that satisfies the following properties.*

- *For every set $e \subset V$ of size at least 2,*

$$\Pr(\omega(u) \neq \omega(v) \text{ for some } u, v \in e) \leq O(\beta^{-1/2}\sqrt{\log|e|}) \times \max_{u,v\in e} \|\varphi(\bar{u}) - \varphi(\bar{v})\|.$$

- *For every u and v such that $\|\varphi(\bar{u}) - \varphi(\bar{v})\|^2 \geq \beta$, $\Pr(\omega(u) \neq \omega(v)) \geq 0.3$.*

Proof. We sample a random Gaussian vector $g \sim \mathcal{N}(0, I_n)$ (each component g_i of g is distributed as $\mathcal{N}(0,1)$, all random variables g_i are mutually independent). Let N be a Poisson process on $\mathbb{R}$ with rate $1/\sqrt{\beta}$. Let $w(u) = 1$ if $N(\langle g, u\rangle)$ is even, and $w(u) = 0$ if $N(\langle g, \varphi(\bar{u})\rangle)$ is odd. Note that $\omega(u) = \omega(v)$ if and only if $N(\langle g, \varphi(\bar{u})\rangle) - N(\langle g, \varphi(\bar{v})\rangle)$ is even.

Consider a set $e \subset V$ of size at least 2. Denote $\operatorname{diam}(e) = \max_{u,v\in e} \|\varphi(\bar{u}) - \varphi(\bar{v})\|$. Let $\tau_m = \min_{w\in e}\langle g, \varphi(\bar{w})\rangle$ and $\tau_M = \max_{w\in e}\langle g, \varphi(\bar{w})\rangle$. Note that

$$N(\tau_m) = \min_{w\in e} N(\langle g, \varphi(\bar{w})\rangle) \qquad \text{and} \qquad N(\tau_M) = \max_{w\in e} N(\langle g, \varphi(\bar{w})\rangle).$$

If all numbers $N(\langle g, \varphi(\bar{u})\rangle)$ are equal then $\omega(u) = \omega(v)$ for all $u, v \in e$. Thus if $\omega(u) \neq \omega(v)$ for some $u, v \in e$ then $N(\langle g, \varphi(\bar{u})\rangle) \neq N(\langle g, \varphi(\bar{v})\rangle)$ for some $u, v \in e$. In particular, then $N(\tau_M) - N(\tau_m) > 0$. Given g, $N(\tau_M) - N(\tau_m)$ is a Poisson random variable with rate $(\tau_M - \tau_m)/\sqrt{\beta}$. We have,

$$\begin{aligned}\Pr\left(\omega(u) \neq \omega(v) \text{ for some } u, v \in e \mid g\right) &\leq \Pr\left(N(\tau_M) - N(\tau_m) > 0 \mid g\right) \\ &= 1 - e^{-(\tau_M - \tau_m)/\sqrt{\beta}} \leq \beta^{-1/2}(\tau_M - \tau_m).\end{aligned}$$

Let $\xi_{uv} = \langle g, \varphi(\bar{u})\rangle - \langle g, \varphi(\bar{v})\rangle$ for $u, v \in e$ $(u \neq v)$. Note that ξ_{uv} are Gaussian random variables with mean 0, and

$$\operatorname{Var}[\xi_{uv}] = \operatorname{Var}[\langle g, \varphi(\bar{u})\rangle - \langle g, \varphi(\bar{v})\rangle] = \|\varphi(\bar{u}) - \varphi(\bar{v})\|^2 \leq \operatorname{diam}(e)^2$$

Note that the expectation of the maximum of (not necessarily independent) N Gaussian random variables with standard deviation bounded by σ is $O(\sqrt{\log N}\sigma)$. We have,

$$\mathbb{E}\left[\tau_M - \tau_m\right] = \mathbb{E}\left[\max_{u,v \in e}(\xi_{uv})\right] = O(\sqrt{\log |e|}\operatorname{diam}(e))$$

since the total number of random variables ξ_{uv} is $|e|(|e| - 1)$. Therefore,

$$\begin{aligned}\Pr\left(\omega(u) \neq \omega(v) \text{ for some } u, v \in e\right) &\leq \beta^{-1/2}\,\mathbb{E}\left[\tau_M - \tau_m\right] \\ &= O(\beta^{-1/2}\sqrt{\log |e|}\max_{u,v \in e}\|\varphi(\bar{u}) - \varphi(\bar{v})\|).\end{aligned}$$

We proved that ω satisfies the first property. Now we verify that ω satisfies the second condition. Consider two vertices u and v with $\|\varphi(\bar{u}) - \varphi(\bar{v})\|^2 \geq \beta$. Given g, the random variable $Z = N(\langle g, \varphi(\bar{u})\rangle) - N(\langle g, \varphi(\bar{v})\rangle)$ has Poisson distribution with rate $\lambda = |\langle g, \varphi(\bar{u})\rangle) - \langle g, \varphi(\bar{v})\rangle|/\sqrt{\beta}$. We have,

$$\Pr\left(Z \text{ is even} \mid g\right) = \sum_{k=0}^{\infty}\Pr\left(Z = 2k \mid g\right) = \sum_{k=0}^{\infty}\frac{e^{-\lambda}\lambda^{2k}}{(2k)!} = \frac{1 + e^{-2\lambda}}{2}.$$

Note that λ is the absolute value of a Gaussian random variable with mean 0 and standard deviation $\sigma = \|\varphi(\bar{u}) - \varphi(\bar{v})\|/\sqrt{\beta} \geq 1$. Thus $\Pr\left(Z \text{ is even}\right) = \mathbb{E}\left[1 + e^{-2\sigma|\gamma|}\right]/2$, where γ is a standard Gaussian random variable, $\gamma \sim \mathcal{N}(0, 1)$. We have,

$$\Pr\left(\omega(u) \neq \omega(v)\right) = \mathbb{E}\left[\frac{1 - e^{-2\sigma|\gamma|}}{2}\right] \geq \mathbb{E}\left[\frac{1 - e^{-2|\gamma|}}{2}\right] \geq 0.3.$$

◀

Now we use the algorithm from Theorem 10 to obtain ℓ_2–ℓ_2^2 hypergraph orthogonal separators. The only difference is that we use the procedure from Lemma 18 rather than from Lemma 13 to generate assignments ω. We obtain a ℓ_2–ℓ_2^2 hypergraph orthogonal separator.

▶ **Theorem 19.** *Random set S is a hypergraph m-orthogonal separator with distortion*

$$D_{\ell_2^2} = O(m) \qquad \text{and} \qquad D_{\ell_2}(r) = O(\beta^{-1/2}\sqrt{\log r}\, m \log m \log\log m),$$

probability scale $\alpha \geq 1/n$ and separation threshold $\beta \in (0, 1)$.

Proof. The proof of the theorem is almost identical to that of Theorem 15. We first check conditions 1 and 2 of ℓ_2–ℓ_2^2 hypergraph orthogonal separators in the same way as we checked conditions 1 and 2 of hypergraph orthogonal separators in Theorem 15. When we verify that property 3 holds, we use bounds from Lemma 18. The only difference is how we upper bound the probability of the event $\mathcal{E}_2$.

If $\mathcal{E}_2$ happens then (1) $r \le \rho_m$ (since $A = e$) and (2) $W(u) \neq W(v)$ for some $u, v \in e$. The probability that $r \le \rho_m$ is ρ_m. We upper bound the probability that $W(u) \neq W(v)$ for some $u, v \in e$. For each $i \in \{1, \dots, l\}$,

$$\Pr\left(\tilde{\omega}_i(u) \neq \tilde{\omega}_i(v) \text{ for some } u, v \in e\right) \le O(\beta^{-1/2}\sqrt{\log|e|}\,\log\log m) \max_{u,v\in e} \|\varphi(\bar{u}) - \varphi(\bar{v})\|$$

$$\le O(\beta^{-1/2}\sqrt{\log|e|}\,\log\log m) \max_{u,v\in e} \frac{\|\bar{u}-\bar{v}\|}{\min(\|\bar{u}\|, \|\bar{v}\|)}$$

$$\le O(\beta^{-1/2}\sqrt{\log|e|}\,\log\log m) \times \rho_m^{-1/2} \times \max_{u,v\in e} \|\bar{u}-\bar{v}\|.$$

By the union bound over $i \in \{1, \dots, l\}$, the probability that $W(u) \neq W(v)$ for some $u, v \in e$ is at most $O(l \times \beta^{-1/2}\sqrt{\log|e|}\,\log\log m) \times \rho_m^{-1/2} \times \max_{u,v\in e} \|\bar{u}-\bar{v}\|$. Therefore,

$$\Pr(\mathcal{E}_2) \le \rho_m \times O(l \times \beta^{-1/2}\sqrt{\log|e|}\,\log\log m) \times \rho_m^{-1/2} \times \max_{u,v\in e} \|\bar{u}-\bar{v}\|$$

$$\le O(l \times \beta^{-1/2}\sqrt{\log|e|}\,\log\log m) \times \rho_m^{1/2} \times \max_{u,v\in e} \|\bar{u}-\bar{v}\|.$$

We get that the probability that e is cut by S is at most

$$\Pr(\mathcal{E}_1) + \Pr(\mathcal{E}_2) \le \max_{u,v\in e} \|\bar{u}-\bar{v}\|^2 + O(l \times \beta^{-1/2}\sqrt{\log|e|}\,\log\log m)\,\rho_m^{1/2} \max_{u,v\in e} \|\bar{u}-\bar{v}\|$$

$$\le \max_{u,v\in e} \|\bar{u}-\bar{v}\|^2 + O(l \times \beta^{-1/2}\sqrt{\log|e|}\,\log\log m) \min_{w\in e} \|\bar{w}\| \max_{u,v\in e} \|\bar{u}-\bar{v}\|.$$

For $D_{\ell_2^2} = 1/\alpha$ and $D_{\ell_2}(r) = O(\beta^{-1/2}\sqrt{\log r}\,\log m \log\log m)/\alpha$, we get

$$\Pr(e \text{ is cut by } S) \le \alpha D_{\ell_2^2} \cdot \max_{u,v\in e} \|\bar{u}-\bar{v}\|^2 + \alpha D_{\ell_2}(|e|) \cdot \min_{w\in e} \|\bar{w}\| \cdot \max_{u,v\in e} \|\bar{u}-\bar{v}\|.$$

Note that $\alpha \ge 1/2^l \ge \Omega(1/m)$. Thus

$$D_{\ell_2^2} = O(m) \qquad \text{and} \qquad D_{\ell_2}(r) = O(\beta^{-1/2}\sqrt{\log r}\, m \log m \log\log m).$$

◀

6 Algorithm for Hypergraph Small Set Expansion via ℓ_2–ℓ_2^2 Hypergraph Orthogonal Separators

In this section, we present another algorithm for Hypergraph Small Set Expansion. The algorithm finds a set with expansion proportional to $\sqrt{\phi^*_{G,\delta}}$. The proportionality constant depends on degrees of vertices and hyperedge size but not on the graph size. Here, we present our result for arbitrary hypergraphs. The result for uniform hypergraphs (Theorem 4) stated in the introduction follows from our general result. In order to state our result for arbitrary graphs, we need the following definition.

▶ **Definition 20.** Consider a hypergraph $H = (V, E)$. Suppose that for every edge e we are given a non-empty subset $e^\circ \subseteq e$. Let

$$\eta(u) = \sum_{e: u\in e^\circ} \frac{\log_2|e|}{|e^\circ|} \qquad \text{and} \qquad \eta_{max} = \max_{u\in V} \eta(u).$$

Finally, let $\eta^H_{\max}$ be the minimum of η_{max} over all possible choices of subsets e°.

▶ **Claim 6.1.** 1. $\eta^H_{max} \le \max_{u\in V} \sum_{e:u\in e}(\log_2 |e|)/|e|$.
2. *If H is a r-uniform graph with maximum degree d_{max} then $\eta^H_{max} \le (d_{max}\log_2 r)/r$.*
3. *Suppose that we can choose one vertex in every edge so that no vertex is chosen more than once. Then $\eta^H_{max} \le \log_2 r_{max}$, where r_{max} is the size of the largest hyperedge in H.*

Proof.

1. Let $e^\circ = e$ for every $e \in E$. We have, $\eta^H_{\max} \le \max_{u\in V} \sum_{e:u\in e}(\log_2 |e|)/|e|$.
2. By **1**, $\eta^H_{\max} \le \max_{u\in V} \sum_{e:u\in e}(\log_2 |e|)/|e| = \max_{u\in V} \sum_{e:u\in e}(\log_2 r)/r = (d_{\max}\log_2 r)/r$.
3. For every edge $e \in E$, let e° be the set that contains the vertex chosen for e. Then $|e^\circ| = 1$ and $|\{e : u \in e^\circ\}| \le 1$ for every u. We have,

$$\eta^H_{\max} \le \max_{u\in V} \sum_{e:u\in e^\circ} \frac{\log_2 |e|}{|e^\circ|} \le \max_{u\in V} \sum_{e:u\in e^\circ} \frac{\log_2 r_{\max}}{1} = \log_2 r_{\max}.$$

◀

▶ **Theorem 21.** *There is a randomized polynomial-time algorithm for the Hypergraph Small Set Expansion problem that given a hypergraph $H = (V, E)$, and parameters $\varepsilon \in (0, 1)$ and $\delta \in (0, 1/2]$, finds a set $S \subset V$ of size at most $(1 + \varepsilon)\delta n$ such that*

$$\begin{aligned}\phi(S) &\le O_\varepsilon\left(\delta^{-1}\log\delta^{-1}\log\log\delta^{-1}\sqrt{\eta^H_{max}\cdot\phi^*_{H,\delta}} + \delta^{-1}\phi^*_{H,\delta}\right)\\ &= \tilde{O}_\varepsilon\left(\delta^{-1}\left(\sqrt{\eta^H_{max}\phi^*_{H,\delta}} + \phi^*_{H,\delta}\right)\right),\end{aligned}$$

In particular, if H is an r-uniform hypergraph with maximum degree d_{max}, then we have,

$$\phi(S) \le \tilde{O}_\varepsilon\left(\delta^{-1}\left(\sqrt{d_{max}\frac{\log_2 r}{r}\phi^*_{H,\delta}} + \phi^*_{H,\delta}\right)\right).$$

Proof. The proof is similar to that of Theorem 11. We solve the SDP relaxation for H-SSE and obtain an SDP solution $\{\bar{u}\}$. Denote the SDP value by sdp-cost. Consider an ℓ_2–ℓ_2^2 hypergraph orthogonal separator S with $m = 4/(\varepsilon\delta)$ and $\beta = \varepsilon/4$. Define a set S':

$$S' = \begin{cases} S, \text{ if } |S| \le (1+\varepsilon)\delta n,\\ \varnothing, \text{ otherwise.}\end{cases}$$

Clearly, $|S'| \le (1+\varepsilon)\delta n$. As in the proof of Theorem 11, $\Pr(u \in S') \in \left[\frac{\alpha}{2}\|\bar{u}\|^2, \alpha\|\bar{u}\|^2\right]$. Note that

$$\Pr(S' \text{ cuts edge } e) \le \Pr(S \text{ cuts edge } e) \le \alpha D_{\ell_2}\max_{u,v\in e}\|\bar{u}-\bar{v}\|^2 + \alpha D_{\ell_2}(r)\min_{w\in e}\|\bar{w}\|\max_{u,v\in e}\|\bar{u}-\bar{v}\|.$$

Let $\mathcal{C} = \alpha^{-1}\mathbb{E}[|E_{cut}(S')|]$. Let $Z = |S'| - \frac{|E_{cut}(S')|}{4\mathcal{C}}$. We have,

$$\mathbb{E}[Z] = \mathbb{E}[|S'|] - \mathbb{E}\left[\frac{|E_{cut}(S')|}{4\mathcal{C}}\right] \ge \sum_{u\in V}\frac{\alpha}{2}\cdot\|\bar{u}\|^2 = \frac{\alpha}{2} - \frac{\alpha}{4} = \frac{\alpha}{4}.$$

Now we upper bound $\mathcal{C}$. Consider the optimal choice of e° for H in the definition of $\eta^H_{\max}$.

$$\begin{aligned}
\mathcal{C} &= \alpha^{-1}\mathbb{E}\left[|E_{cut}(S')|\right] \le \alpha^{-1}\sum_{e\in E}\Pr\left(e \text{ is cut by } S\right)\\
&\le D_{\ell_2^2}\sum_{e\in E}\max\|\bar{u}-\bar{v}\|^2 + \sum_{e\in E}D_{\ell_2}(|e|)\min_{w\in e}\|\bar{w}\|\max_{u,v\in e}\|\bar{u}-\bar{v}\|\\
&\le D_{\ell_2^2}\cdot\text{sdp-cost} + \sum_{e\in E}D_{\ell_2}(|e|)\sum_{w\in e^\circ}\left(\frac{\|\bar{w}\|}{|e^\circ|}\right)\times\max_{u,v\in e}\|\bar{u}-\bar{v}\|\\
&\le D_{\ell_2^2}\cdot\text{sdp-cost} + \sum_{e\in E}\sum_{w\in e^\circ}\frac{D_{\ell_2}(|e|)\|\bar{w}\|}{\sqrt{|e^\circ|}}\times\frac{\max_{u,v\in e}\|\bar{u}-\bar{v}\|}{\sqrt{|e^\circ|}}\\
&\le D_{\ell_2^2}\cdot\text{sdp-cost} + \sqrt{\sum_{e\in E}\sum_{w\in e^\circ}\frac{D_{\ell_2}(|e|)^2\|\bar{w}\|^2}{|e^\circ|}}\sqrt{\sum_{e\in E}\sum_{w\in e^\circ}\frac{\max_{u,v\in e}\|\bar{u}-\bar{v}\|^2}{|e^\circ|}}\\
&\le D_{\ell_2^2}\cdot\text{sdp-cost} + \sqrt{\sum_{w\in V}\sum_{e:w\in e^\circ}\frac{D_{\ell_2}(|e|)^2}{|e^\circ|}\|\bar{w}\|^2}\,\sqrt{\text{sdp-cost}}.
\end{aligned}$$

For every vertex w,

$$\sum_{e:w\in e^\circ}\frac{D_{\ell_2}(|e|)^2}{|e^\circ|} \le O_\beta(m\log m\log\log m)^2\sum_{e:w\in e^\circ}\frac{\log_2|e|}{|e^\circ|} \le O_\beta(m\log m\log\log m)^2\times\eta^H_{\max}.$$

and $\sum_{w\in V}\|\bar{w}\|^2 = 1$. Therefore, $\mathcal{C} \le O_\beta\left(m\,\text{sdp-cost} + m\log m\log\log m\,\sqrt{\eta^H_{\max}\cdot\text{sdp-cost}}\right)$. By the argument from Theorem 11, we get that if we sample S' sufficiently many times (i.e., $(4n^2/\alpha)$ times), we will find a set S' such that

$$\frac{|E_{cut}(S')|}{|S'|} \le 4\mathcal{C} \le O_\beta\left(\delta^{-1}\log\delta^{-1}\log\log\delta^{-1}\sqrt{\eta^H_{\max}\cdot\text{sdp-cost}} + \delta^{-1}\,\text{sdp-cost}\right)$$

with probability exponentially close to 1. ◀

7 SDP Intgrality Gap

In this section, we present an integrality gap for the SDP relaxation for H-SSE. We also give a lower bound on the distortion of a hypergraph m-orthogonal separator.

▶ **Theorem 22.** *For $\delta = 1/r$, the integrality gap of the SDP for H-SSE is at least $1/(2\delta) = r/2$.*

Proof. Consider a hypergraph $H = (V, E)$ on $n = r$ vertices with one hyperedge $e = V$ (e contains all vertices). Note that the expansion of every set of size $\delta n = 1$ is 1. Thus $\phi^*_{H,\delta} = 1$.

Consider an SDP solution that assigns vertices mutually orthogonal vectors of length $1/\sqrt{r}$. It is easy to see this is a feasible SDP solution. Its value is $\max_{u,v\in e}\|\bar{u}-\bar{v}\|^2 = 2/r$. Therefore, the SDP integrality gap is at least $r/2$. ◀

Now we give a lower bound on the distortion of hypergraph m-orthogonal separators.

▶ **Lemma 23.** *For every $m > 4$, there is an SDP solution such that every hypergraph m-orthogonal separator with separation threshold $\beta \ge 0$ has distortion at least $\lceil m\rceil/4$.*

Proof. Consider the SDP solution from Theorem 22 for $n = r = \lceil m\rceil$. Consider a hypergraph m-orthogonal separator S for this solution. Let D be its distortion. Note that condition (2) from the definition of hypergraph orthogonal separators applies to any pair of distinct vertices (u, v) since $\langle\bar{u},\bar{v}\rangle = 0$.

By the inclusion–exclusion principle, we have,

$$\begin{aligned}\Pr(|S|=1) &\geq \sum_{u\in S}\Pr(u\in S) - \frac{1}{2}\sum_{u,v\in S, u\neq v}\Pr(u\in S, v\in S)\\ &\geq \sum_{u\in S}\alpha\|\bar{u}\|^2 - \frac{1}{2}\sum_{u,v\in S,u\neq v}\frac{\alpha\min(\|\bar{u}\|^2,\|\bar{v}\|^2)}{m} = \alpha - \frac{\alpha n(n-1)}{2mr} \geq \alpha/2.\end{aligned}$$

On the other hand, if $|S|=1$ then S cuts e. We have,

$$\Pr(|S|=1) \leq \Pr(S \text{ cuts } e) \leq \alpha D \max_{u,v\in e}\|\bar{u}-\bar{v}\|^2 = 2\alpha D/r.$$

We get that $\alpha/2 \leq 2\alpha D/r$ and thus $D \geq r/4 = \lceil m\rceil/4$. ◀

8 Reduction from Vertex Expansion to Hypergraph Expansion

In the reduction from vertex expansion to hypergraph expansion, we will use the notion of *Symmetric Vertex Expansion*. For a graph $G=(V,E)$, and for a set $S\subset V$, we define its outer neighborhood $N(S)$ as follows.

$$N(S) = \{u\in\bar{S} : \exists\, v\in S \text{ such that } \{u,v\}\in E\}.$$

The symmetric vertex expansion of a set, denoted by $\Phi^V(S)$, is defined as

$$\Phi^V(S) = \frac{|N(\bar{S})\cup N(S)|}{\min(|S|,|\bar{S}|)} \qquad \text{and} \qquad \Phi^V_{G,\delta} = \min_{\substack{S\subset V\\ 0<|S|\leq\delta n}}\Phi^V(S).$$

We will use the following reduction from vertex expansion to symmetric vertex expansion.

▶ **Theorem 24** (Louis, Raghavendra and Vempala [13]). *Given a graph G, there exists a graph G' such that $c_1\phi^V_{G,\delta} \leq \Phi^V_{G',\delta} \leq c_2\phi^V_{G,\delta}$. where $c_1, c_2 > 0$ are absolute constants, and the maximum degree of graph G' is equal to the maximum degree of graph G. Moreover, there exists a polynomial time algorithm to compute such graph G'.*

Proof of Theorem 6. Starting with graph G, we use Theorem 24 to obtain a graph $G'=(V',E')$ such that $c_1\phi^V_{G,\delta} \leq \Phi^V_{G',\delta} \leq c_2\phi^V_{G,\delta}$. Next we construct hypergraph $H=(V',E'')$ as follows. For every vertex $v\in V'$, we add the hyperedge $\{v\}\cup N(\{v\})$ to E'' (note that $N(\{v\})$ is the set of neighbors of v in G). Fix an arbitrary set $S\subset V$.

We first show that $\Phi^V(S)\leq\phi_H(S)$. Consider the vertices $N(\bar{S})$. Each vertex in $v\in N(\bar{S})$ has a neighbor, say u, in $\bar{S}$. Therefore the hyperedge $\{v\}\cup N(\{v\})$ is cut by S in H. Similarly, for each vertex $v\in N(S)$, the hyperedge $\{v\}\cup N(\{v\})$ is cut by S in H. All these hyperedges are disjoint by construction. Therefore, $\Phi^V(S) = \frac{|N(\bar{S})|+|N(S)|}{|S|} \leq \frac{|E_{cut}(S)|}{|S|} \leq \phi_H(S)$.

Now we verify that $\phi_H(S)\leq\Phi^V(S)$. For any hyperedge $(\{v\}\cup N(\{v\}))\in E_{cut}(S)$, the vertex v has to belong to either $N(\bar{S})$ or $N(S)$. Therefore, $\phi_H(S)\leq\frac{|E_{cut}(S)|}{|S|}\leq\frac{|N(\bar{S})|+|N(S)|}{|S|} = \Phi^V(S)$. Thus, we get that $\phi_H(S)=\Phi^V(S)$ for every $S\subset V$, and hence $\phi^*_{H,\delta} = \Phi^V_{G',\delta}$. Therefore, $c_1\phi^V_{G,\delta}\leq\phi^*_{H,\delta}\leq c_2\phi^V_{G,\delta}$.

Finally, we upper bound $\eta^H_{\max}$. We use part 3 of Claim 6.1. We choose vertex v in the hyperedge $\{v\}\cup N(\{v\})$. By Claim 6.1, $\eta^H_{\max}\leq\log_2 r_{\max}$, where $r_{\max}$ is the size of the largest hyperedge. Note that $|\{v\}\cup N(\{v\})| = \deg v+1$. Thus $\eta^H_{\max}\leq\log_2 r_{\max}\leq\log_2(d_{\max}+1)$ ◀

References

1 S. Arora, and R. Ge. New tools for graph coloring. APPROX 2011.
2 S. Arora, J. R. Lee, and A. Naor. Euclidean distortion and the sparsest cut. STOC 2005.
3 S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings and graph partitioning. STOC 2004.
4 N. Bansal, U. Feige, R. Krauthgamer, K. Makarychev, V. Nagarajan, J. Naor, and R. Schwartz. Min-max Graph Partitioning and Small Set Expansion. FOCS 2011.
5 U. Catalyurek, and C. Aykanat. Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication. IEEE Trans. Parallel Distrib. Syst. 1999.
6 E. Chlamtac, K. Makarychev, and Y. Makarychev. How to Play Unique Games Using Embeddings. FOCS 2006.
7 K. Devine, E. Boman, R. Heaphy, R. Bisseling and U. Catalyurek. Parallel hypergraph partitioning for scientific computing. IPDPS 2006.
8 G. Karypis, R. Aggarwal, V. Kumar and S. Shekhar. Multilevel hypergraph partitioning: applications in VLSI domain. IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 1999.
9 A. Louis and K. Makarychev. Approximation Algorithm for Sparsest k-Partitioning. SODA 2014.
10 A. Louis and Y. Makarychev. Approximation Algorithms for Hypergraph Small Set Expansion and Small Set Vertex Expansion. arXiv:1404.4575 [cs.DS].
11 A. Louis. Hypergraph Markov Operators, Eigenvalues and Approximation Algorithms. Manuscript 2014.
12 A. Louis, P. Raghavendra and S. Vempala. Private Communication. 2012.
13 A. Louis, P. Raghavendra and S. Vempala. The Complexity of Approximating Vertex Expansion. FOCS 2013.
14 K. Makarychev and Y. Makarychev. Nonuniform Graph Partitioning with Unrelated Weights. To appear at ICALP 2014.
15 P. Raghavendra and D. Steurer. Graph expansion and the unique games conjecture. STOC 2010.
16 P. Raghavendra, D. Steurer, and P. Tetali. Approximations for the isoperimetric and spectral profile of graphs and related parameters. STOC 2010.

Robust Appointment Scheduling

Shashi Mittal[1], Andreas S. Schulz[2], and Sebastian Stiller[3]

1 **Amazon.com, Inc., Seattle, WA, USA**
`mshashi@alum.mit.edu`
2 **Massachusetts Institute of Technology, Cambridge, MA, USA**
`schulz@mit.edu`
3 **TU Berlin, Berlin, Germany**
`stiller@math.tu-berlin.de`

Abstract

Health care providers are under tremendous pressure to reduce costs and increase quality of their services. It has long been recognized that well-designed appointment systems have the potential to improve utilization of expensive personnel and medical equipment and to reduce waiting times for patients. In a widely influential survey on outpatient scheduling, Cayirli and Veral (2003) concluded that the "biggest challenge for future research will be to develop easy-to-use heuristics." We analyze the appointment scheduling problem from a robust-optimization perspective, and we establish the existence of a closed-form optimal solution–arguably the simplest and best 'heuristic' possible. In case the order of patients is changeable, the robust optimization approach yields a novel formulation of the appointment scheduling problem as that of minimizing a concave function over a supermodular polyhedron. We devise the first constant-factor approximation algorithm for this case.

1998 ACM Subject Classification G.2.1 Combinatorics – Combinatorial Algorithms, G.1.6 Optimization – Nonlinear programming, I.1.2 Algorithms – Analysis of Algorithms

Keywords and phrases Robust Optimization, Health Care Scheduling, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.356

1 Introduction

We study the problem of appointment scheduling in a robust optimization framework. The appointment scheduling problem arises in many service operations where customers are served sequentially, service times of customers are uncertain, and one needs to assign time slots for serving the customers in advance. Arguably the most relevant practical setting where this problem arises is in health care services. Modern health care involves the usage of several high-cost devices and facilities such as MRI installations, CT scanners and operation rooms, in addition to highly trained and well-paid personnel. For health care providers, appointment scheduling is vital to ensure a high utilization of their resources as well as a high quality of service.[1] For example, consider the problem of scheduling surgeries for outpatients in an operation room at a hospital. The information about which surgeries are to be performed on a particular day is known in advance. However, the time needed to perform each surgery can vary. Typically on the preceding day, the hospital manager needs to decide the time at which a particular surgery is scheduled to start, and how much time to

[1] Excessive waiting times are often the major reason for patients' dissatisfaction in outpatient services. Nowadays, reasonable waiting times are expected in addition to clinical competence [5].

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 356–370

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

assign to that surgery. If the manager assigns a rather small time interval for a surgery, then it is more likely that the actual time of the surgery will exceed its assigned duration, thus delaying the next surgery. The inconvenience and costs resulting from the delay of both the patients and the staff constitute the *overage* cost of that surgery. If on the other hand, the hospital manager assigns an excessively long interval to a particular surgery, then chances are that the surgery may end early and the operation room will be left idle until the next surgery commences. In that case, the hospital incurs *underage* cost, which corresponds to the under-utilization of the resources in the operation room. Therefore, an ideal appointment schedule should achieve the right trade-off between the underage and the overage costs.

Existing models in the literature for the appointment scheduling problem include queueing models [20, 21], continuous stochastic models [6, 13, 17] and discrete stochastic models [1, 2]. In the stochastic models, the processing times of the jobs[2] are assumed to be independent random variables, and the objective is to find an appointment schedule that minimizes the expected cost. In all these models, one assumes complete knowledge about the distribution of the processing times of the jobs. However, in many service settings the distributions may not be known accurately, limiting the utility of the stochastic models. There might not be sufficient historical data of the processing times of the jobs to get a reasonable estimate of the probability distributions. Furthermore, because the cost function in the stochastic model is the expectation of a non-linear function of several random variables, the computational cost of finding an optimal schedule is significantly high. As a consequence, the methods employed to solve the problem are usually based on heuristics with no provable bounds on the running time of the algorithm or on the quality of the solutions they generate. Other methods require the use of advanced techniques such as Monte-Carlo simulations or submodular function minimization. Such techniques may not necessarily be practical in many situations. The drawbacks of the stochastic models mentioned above are not limited to the appointment scheduling problem, but are encountered in many situations in which there is uncertainty. Robust optimization is an alternative framework to address uncertainty. In robust optimization, the uncertainty in the input parameters is handled using uncertainty sets instead of random variables (see e. g. [3, 4]). Robust optimization models are often more tractable when compared to the corresponding stochastic optimization models.

Robust optimization models appear to be particularly useful in health care, as they attempt to reduce the cost and stress level of a bad day, rather than taking into account only the average. For example, stochastic models often give a "dome shaped" schedule, reckoning on the small probability of delays accumulating in the morning. But in case of several early delays such a schedule incurs a high cost as these delays spill over on a large part of the day. A robust schedule is not "dome shaped," but assigns some slack time in the morning as well.

The main contributions of this paper can be summarized as follows.

Robust Formulation of the Problem. We propose to look at the appointment scheduling problem in a robust optimization framework. For each job we only need the following information: the minimum and the maximum possible time the job will take to complete, the underage cost if the job finishes early, and the overage cost if the job finishes late. The objective in the robust model is to find a schedule for which the cost in the worst-case scenario of the realized processing times of the jobs is minimized. For simplicity of presentation, we

[2] From now on, we use the term 'job' to refer to the kind of service provided in any one specific context; for instance, in the health care setting, a job could correspond to a surgery.

focus on the important case where all underage cost coefficients are equal, i. e., they represent the opportunity cost of the facility.

A Closed-form Solution. We propose an intuitive method for scheduling jobs that aims to balance the maximum possible underage cost of a job with the maximum possible overage cost due to that job. This approach yields an optimal solution to the robust model. Its biggest advantage is that it gives a simple, easy to compute, closed-form solution for the optimal duration assigned to each job. Unlike existing, stochastic methods for appointment scheduling, our solution is simple enough that it can be implemented in a spreadsheet, thus greatly enhancing its practical value.

Ordering Problem. In health care, one finds both, applications in which the ordering is fixed, and applications where reshuffling jobs is possible to further reduce the cost [5]. Despite the relevance of the case where reordering jobs is possible, most research on methods to solve appointment scheduling neglect the optimization potential of ordering. One reason may be that most methods are already involved even without considering the ordering problem. We show that the closed-form expression for the optimal solution for the fixed-ordering problem can be used to derive constant-factor approximation algorithms for the ordering problem.

Related Work

An overview of the appointment scheduling problem is given in [5]. The existing literature on appointment scheduling can roughly be divided into three categories: queueing models, stochastic optimization models and stochastic models which use notions of discrete convexity, for example, submodular functions over an integer lattice. We discuss the relevant literature for all three models below.

Wang [20] proposes a queueing model for the problem, in which the processing times of the jobs are assumed to be independent and identically distributed random variables with exponential distribution. Both static and dynamic problems (i. e. the case when all the information about the jobs is not known in advance) are considered in this model, and an optimal schedule is obtained by solving a set of non-linear equations. In [21], the model is generalized to the case where the jobs can have different mean processing times. For this model, it is shown that the optimal sequence of the execution of the jobs is to process them in the increasing order of their mean processing times.

Denton and Gupta [6] formulate the problem as a two-stage stochastic linear program, and then use a sequential bounding algorithm to solve the corresponding stochastic optimization problem. They also give general upper bounds on the cost of a schedule which do not depend on the particular distribution of the processing times or the cost parameters of the jobs. Robinson and Chen [17] use a Monte Carlo integration technique to compute near-optimal solutions for the appointment scheduling problem. They show that an optimal schedule for this model has a "dome shaped" structure. That is, the allowances for the assigned durations for the jobs first increase, and then decrease steadily for jobs in the end of the sequence. They also give heuristics which approximate this dome shaped structure of the optimal schedule. Green et al. [8] consider the problem of outpatient appointment scheduling in which serving emergency patients is also permitted. They formulate the problem as a dynamic stochastic control problem and establish properties of an optimal policy for real-time scheduling and capacity allocation. Yet another way of computing an appointment schedule is using local search: Kandoorp and Koole [13] show that a local search algorithm converges to an optimal schedule. Gupta [9] considers the problem of optimally sequencing two jobs,

and establishes the optimality of an ordering when a stochastic dominance condition holds for the distribution of the processing durations of the two jobs.

More recently, Begen and Queyranne [2] show that when the processing times of the jobs are discrete random variables with finite integer support, then there is an optimal schedule which is integral (i. e. the assigned starting times of the jobs have integer values in the optimal solution). They also show that under very general conditions, the cost function with respect to an integer appointment schedule is submodular. An optimal solution can then be found using well known algorithms for submodular function minimization (e. g. [12, 15]). This idea has also been extended to a get a near-optimal schedule for a data driven model [1], where the processing time distributions of the jobs are not known in advance, but instead one uses the past data of the realized processing times of the jobs to approximate the distributions.

2 Appointment Scheduling and Ordering

The appointment scheduling problem consists of determining starting times for a fixed sequence of jobs with uncertain processing times. When the appointment schedule is executed and the exact processing times materialize, each jobs starts at its planned appointment time or at the completion time of its predecessor, whichever comes later. The goal is to minimize a weighted sum of idle times of the server and delays of the jobs. In the scheduling and ordering version of the problem, one also decides on the order in which the jobs are processed.

Formally, instances of both the *robust appointment scheduling and ordering problem* and the *robust appointment scheduling problem* are given by an n-tuple of jobs and a positive value u, the cost per unit of idle time. Each job i is characterized by $[\underline{p}_i, \overline{p}_i]$, an interval of (non-negative) possible processing times, and a positive value o_i, the per unit-time cost for the delay of job i.

A solution to the robust appointment scheduling problem is a vector of appointments $A = (A_1, \ldots, A_{n+1})$. Here, $A_1 = 0$ is the scheduled start time of job 1, A_{n+1} is the planned completion time of job n, and A_i is the planned completion time of job $i-1$ (and, simultaneously, the planned start time of job i), for all $2 \leq i \leq n$. A solution (π, A) to the robust appointment scheduling and ordering problem is a permutation $\pi : [n] \to [n]$ of the jobs' indices, and an appointment vector for this permutation as above, i. e., $(A_{\pi^{-1}(1)}, \ldots, A_{\pi^{-1}(n+1)})$.[3]

A scenario is any vector $P = (p_1, \ldots, p_n)$ of processing times within the given intervals: $p_i \in [\underline{p}_i, \overline{p}_i]$. We call $\mathbb{P} = \prod_{i=1}^{n} [\underline{p}_i, \overline{p}_i]$ the scenario set. In the following definitions we focus on the scheduling and ordering problem. Fixing the permutation π to the identity (i. e., ignoring it in the formulas) gives the definitions for completion time and cost in the version of the problem without ordering.

For a solution (π, A), the completion time of job $\pi^{-1}(i)$ in a scenario P is

$$C_{\pi^{-1}(i)} = \max(A_{\pi^{-1}(i)}, C_{\pi^{-1}(i-1)}) + p_{\pi^{-1}(i)}.$$

The cost of a solution (π, A) in a scenario P is

$$F\big((\pi, A), P\big) = \sum_{i=1}^{n} \max\left(u(A_{\pi^{-1}(i+1)} - C_{\pi^{-1}(i)}), o_{\pi^{-1}(i)}(C_{\pi^{-1}(i)} - A_{\pi^{-1}(i+1)})\right).$$

[3] As usual, $[n]$ denotes the set $\{1, 2, \ldots, n\}$. In a slight abuse of notation, $A_{\pi^{-1}(n+1)}$ refers to the planned completion time of the last job.

We call the first argument of the maximum the *underage cost*, and the second the *overage cost*. We say a job is *underaged*, respectively, *overaged* depending on which of the two terms is positive.[4] The cost of a solution (π, A) is

$$F(\pi, A) := \sup_{P \in \mathbb{P}} F\big((\pi, A), P\big) = \max_{P \in \mathbb{P}} F\big((\pi, A), P\big),$$

where the second equality holds because F is continuous in P [2].

It is helpful to use $a_i := A_{i+1} - A_i$ as a short-hand notation for the time assigned to job i. We will sometimes identify the schedule A by the vector of assigned times $a = (a_1, \ldots, a_n)$. For convenience, let us also introduce the following notation: $o_{\geq i} := \sum_{j \geq i} o_j$ and $\Delta_i := \overline{p}_i - \underline{p}_i$.

The robust appointment model allows for a number of structural and algorithmic insights. In this extended abstract, we focus on two main results:

▶ **Theorem 1.** *The robust appointment scheduling problem has a closed-form optimal solution.*

In addition to the obvious practical implications, Theorem 1 is also essential in deriving the next result.

▶ **Theorem 2.** *The robust appointment scheduling and ordering problem admits a polynomial-time approximation algorithm with constant-factor performance guarantee.*

We first design a $(2 + \epsilon)$-approximation algorithm that is of interest in its own right and provides very good insights into the essential structure of the robust appointment scheduling and ordering problem. We also give a polynomial-time approximation scheme, which is based on a new formulation of the robust appointment scheduling and ordering problem as that of minimizing a concave function over a supermodular polyhedron.

3 Robust Appointment Scheduling (with Fixed Job Order)

In this section, we consider the problem of finding optimal appointment times for the robust appointment scheduling problem when the job order is fixed to $1, 2, \ldots, n$. We prove the optimality of an appointment schedule that is based on the following idea: charge each part of the total cost to the job that caused it. For the total underage cost, responsibilities are clear: in case $A_{i+1} - C_i > 0$, we want to charge job i for $u(A_{i+1} - C_i)$ of the total cost. Note that $A_{i+1} - C_i \leq a_i - \underline{p}_i$.

The overage cost of job i, i. e., $o_i(C_i - A_{i+1}) > 0$, can be caused by job i itself, namely in case $a_i < p_i$, and by jobs j preceding i that initially cause a delay, i. e., $a_j < p_j$, which spills over to i. We want to distribute the cost $o_i(C_i - A_{i+1})$ to those jobs that initially caused it. Note that $C_i - A_{i+1} \leq \sum_{j \leq i} (p_j - a_j)^+$ and, by reordering of summation,

$$\sum_i o_i \sum_{j \leq i} (p_j - a_j)^+ = \sum_j o_{\geq j} (p_j - a_j)^+ \leq \sum_{j \text{ overaged}} o_{\geq j} (\overline{p}_j - a_j).$$

Hence, we can distribute the entire cost of schedule A in any scenario P by attributing to each underaged job i at most $u(a_i - \underline{p}_i)$ of underage cost and to each overaged job i at most $o_{\geq i}(\overline{p}_i - a_i)$ of overage cost. Formally, we have

[4] In the model considered here, the cost caused by the overage of the i-th job in the sequence depends only on that job. All of our results remain to hold true if the overage cost of the i-th jobs is determined by the $(i + 1)$-st job in the sequence.

▶ **Lemma 3.** *For any appointment schedule A and any scenario $P \in \mathbb{P}$, the cost $F(A,P)$ is bounded as follows:*

$$F(A,P) \leq \sum_{i=1}^{n} \max\left(u(a_i - \underline{p}_i), o_{\geq i}(\overline{p}_i - a_i)\right).$$

Note that the expression on the right-hand side does not depend on P.

This is a very pessimistic perspective. Still, our solution is guided by this pessimism: Every job i shall minimize the maximum over its own maximal underage cost, namely $u(a_i - \underline{p}_i)$, and the maximum total increase in overage cost job i can cause: $o_{\geq i}(\overline{p}_i - a_i)$. Setting these two terms equal, i. e., balancing the potential underage cost and the potential responsibility for overage cost, yields the a_i of the *balanced schedule*.

▶ **Definition 4.** For an instance of the robust appointment scheduling problem, the *balanced schedule* A^B is defined by

$$a_i^B = \frac{u\underline{p}_i + o_{\geq i}\overline{p}_i}{u + o_{\geq i}}$$

for all $i = 1, \ldots, n$.

In this section we show that the *balanced schedule* is optimal and determine its cost, which gives a refined version of Theorem 1.

▶ **Theorem 5.** *For any instance of the robust appointment scheduling problem the corresponding balanced schedule A^B is an optimal solution. The cost of the balanced schedule A^B is*

$$F(A^B) = \sum_{i=1}^{n} \frac{uo_{\geq i}\Delta_i}{u + o_{\geq i}} \quad .$$

The pessimistic cost terms balanced in A^B are the actual costs of jobs in scenarios with the following properties: All jobs are either at maximal or minimal length, and if job i is at maximal length, then so are all jobs after job i. We prove Theorem 5 by showing that this set of $n+1$ scenarios is the set of worst-case scenarios for any optimal solution.

In the rest of this section, we will prove Theorem 5. The following intuitive lemma helps to eliminate some pathological cases from further consideration.

▶ **Lemma 6.** *There exists an optimal appointment schedule A for which $\underline{p}_i \leq a_i \leq \overline{p}_i$.*

Proof. The first part of the inequality was proved in [2]. For the second part, suppose that in an optimal solution A, the duration assigned to some job i is greater than $\overline{p}_i$. Let i be the largest index of a job for which this is the case. Let $\delta = A_{i+1} - A_i - \overline{p}_i$. By assumption, $\delta > 0$. Note that we can focus on $i > 1$. Otherwise we could simply reduce each A_j by δ, for $j = 2, 3, \ldots, n$, and the resulting schedule would have lower cost, which is a contradiction to the optimality of A. We claim that changing A_i to $A_i + \delta$ does not increase the cost in any scenario. We consider two cases:
Case 1: Job $i-1$ is underaged. In this situation, job i is underaged as well. If A_i is changed to $A_i + \delta$, then the underage cost of job $i-1$ increases, but the underage cost of job i decreases by the same amount.
Case 2: Job $i-1$ is overaged. Let C_{i-1} be the completion time of job $i-1$. Then $C_{i-1} > A_i$. If $C_{i-1} - A_i \geq \delta$, then increasing A_i to $A_i + \delta$ only decreases the overage cost of job $i-1$,

and changes nothing else. If $C_{i-1} - A_i < \delta$, then after increasing A_i to $A_i + \delta$, job $i-1$ becomes underaged. However, any increase in the underage cost of job $i-1$ is neutralized by the decrease in the underage cost of job i. The net effect is a decrease in the overall cost, as the overage cost that job $i-1$ was incurring in the earlier schedule has disappeared in the new schedule.

As a result, the cost in every scenario either remains the same, or decreases upon increasing A_i by δ. Therefore, the new schedule is still optimal, and the largest index of some job violating the claim of the lemma is now smaller than i. Iterating the argument, if necessary, completes the proof. ◀

From now on, we will focus on optimal appointment schedules that satisfy the condition in Lemma 6. Let $\mathcal{P}$ denote the set of all scenarios P such that $p_i \in \{\underline{p}_i, \overline{p}_i\}$ for all jobs i. We call these scenarios the *extremal scenarios*. We can show constructively that the cost of any scenario does not decrease by shifting it to an extremal scenario. This eventually shows that we can limit our attention to a finite scenario set.

▶ **Lemma 7.** *Let A^* be any schedule that has the lowest worst-case scenario cost, when only the scenarios in the set $\mathcal{P}$ are considered. Then A^* is an optimal appointment schedule.*

For the proof of Lemma 7 we need some preparation.

▶ **Definition 8.** For an appointment schedule A and a realization of the processing times of jobs P, a *chain* is defined as:

- A single job that is not overaged, or
- A sequence of jobs $i, i+1, \ldots, j$, such that jobs i to $j-1$ are overaged and job j is not overaged, or
- A sequence of jobs $i, i+1 \ldots, j$ all of which are overaged and job j is the last job.

Note that, given an appointment schedule A, the actual execution of the jobs for a given realization of the processing times of jobs P is a union of consecutive chains.

▶ **Lemma 9.** *For any given appointment schedule A, there exists a worst-case scenario P such that $p_i \in \{\underline{p}_i, \overline{p}_i\}$ for all jobs i.*

Proof. Let P be a worst-case scenario for the given appointment schedule A. Suppose P does not satisfy $p_i \in \{\underline{p}_i, \overline{p}_i\}$. Let i be the smallest job index for which this property is violated. We will convert P to a new scenario P', such that p'_i is either $\underline{p}_i$ or $\overline{p}_i$, and the cost corresponding to P', $F(A, P')$, is at least $F(A, P)$. For such a job i, there are the following cases to consider:

Case 1: Job i is not overaged. In this case, setting $p'_i = \underline{p}_i$ and $p'_j = p_j$ for all other jobs j results in increasing the cost of job i, and changes nothing else for other jobs. Thus $F(A, P') \geq F(A, P)$.

Case 2: Job i is overaged, and it is the last job. Then setting $p'_i = \overline{p}_i$ and $p'_j = p_j$ for all other jobs j increases the cost of the last job, and the cost of all other jobs remains the same. Thus $F(A, P') \geq F(A, P)$ in this case as well.

Case 3: Job i is overaged, and it is not the last job. Consider the chain which job i is part of in the realization P. Let $i, \ldots, j$ be the sequence of jobs including and following i in this chain. There are three sub-cases to consider:

- Job j is overaged. This means that the chain of which i is a part of is the last chain. In this case, all the jobs from job i onwards are overaged. Let $p'_i = \overline{p}_i$ and $p'_k = p_k$ for all other jobs k. Then the overage cost of all the jobs i and onwards is higher in P' than in P, while the cost of all the other jobs remains unchanged. Therefore $F(A, P') \geq F(A, P)$.

- Job j is underaged and $o_i + \ldots + o_{j-1} \leq u$. Suppose we reduce the processing time of job i by a sufficiently small amount ϵ that keeps the chain which i is a part of intact. Then the overage cost of jobs $i, \ldots, j-1$ decreases by $(o_i + \ldots + o_{j-1})\epsilon$, while the underage cost of job j increases by $u\epsilon$. Since $o_i + \ldots + o_{j-1} \leq u$, the cost in the modified schedule is at least as much as that in the original schedule. If ϵ can be chosen such that $p'_i = \underline{p}_i$, we are done. Otherwise, the chain that i was part of gets split into smaller chains. However, setting $p'_i = \underline{p}_i$ and $p'_k = p_k$ for all other jobs k still results in a realization P' for which $F(A, P') \geq F(A, P)$.
- Job j is underaged and $o_i + \ldots + o_{j-1} > u$. In this case, let $p'_i = \overline{p}_i$, and $p'_k = p_k$ for all other jobs k. It is possible that by doing so, the underage cost of some job $k \geq j$ decreases, and it may even get overaged. Since $o_i + \ldots + o_{j-1} > u$, it follows that $o_i + \ldots + o_{k-1} > u$ as well. So even though the underage cost of job k decreases, it is more than compensated by the increase in the overage cost of jobs $i, \ldots, k-1$. This holds true for any underaged job $k \geq j$. Therefore, the cost in the realization P' is at least as much as that in the realization P.

We can continue this process for each job that is not extremal, and eventually we will obtain a worst-case scenario in which each job is extremal, and whose realized cost is at least as much as that of the original scenario. Thus, for any appointment schedule A, there exists a worst-case scenario P such that $p_i \in \{\underline{p}_i, \overline{p}_i\}$ for all jobs i. ◀

Lemma 9 implies that in order to compute the worst-case scenario cost of an appointment schedule, it suffices to consider extremal scenarios only:

▶ **Lemma 10.** *For a given appointment schedule A, its cost $F(A)$ is given by*

$$F(A) = \max_{P \in \mathcal{P}} F(A, P).$$

In turn, Lemma 10 completes the proof of Lemma 7. Recall that Lemma 7 states that in order to compute an optimal appointment schedule, it suffices to consider the optimal solution over extremal scenarios.

Proof of Lemma 7. Let A^* be a schedule that has the lowest worst-case scenario cost, when only the scenarios in the set $\mathcal{P}$ are considered. Suppose A^* is not an optimal appointment schedule. Let $\hat{A}$ be an optimal appointment schedule. Lemma 10 implies that there exists $\hat{P} \in \mathcal{P}$ such that $F(\hat{A}) = F(\hat{A}, \hat{P})$. We have the following chain of inequalities:

$$\max_{P \in \mathcal{P}} F(\hat{A}, P) = F(\hat{A}, \hat{P}) = F(\hat{A}) < F(A^*) = \max_{P \in \mathcal{P}} F(A^*, P).$$

Thus, we get $\max_{P \in \mathcal{P}} F(\hat{A}, P) < \max_{P \in \mathcal{P}} F(A^*, P)$, which contradicts the optimality of A^* over the scenarios in the set $\mathcal{P}$. ◀

According to Lemma 7, it suffices to consider the 2^n extremal scenarios when one wants to compute an optimal appointment schedule. For a schedule A we denote by $\mathcal{W}_A$ the set of all extremal, worst-case scenarios. We will now determine the elements of $\mathcal{W}_A$ completely for optimal schedules A. The finiteness of the scenario set $\mathcal{W}_A$ allows for ϵ-shifting arguments that facilitate the derivation of the following two lemmata.

▶ **Lemma 11.** *For any appointment schedule A, if job i is tight or underaged (i. e. $C_i \leq A_{i+1}$) in a worst-case scenario P, then $p_i = \underline{p}_i$.*

Proof. If job i is tight or underaged in a worst-case scenario P and the job is not at minimal length, then reducing the length of i will give a higher cost, contradicting P being a worst-case scenario. ◀

▶ **Lemma 12.** *For an optimal appointment schedule A, and for each job i there is at least one $P \in \mathcal{W}_A$ where job i is not underaged, and at least one $P' \in \mathcal{W}_A$ where job i is not overaged.*

Proof. Suppose job i is underaged for all scenarios in $\mathcal{W}_A$. As $\mathcal{W}_A$ is finite and u is positive, there is $\epsilon > 0$ such that decreasing a_i by ϵ gives a better solution, which is a contradiction. The proof for the other case is similar. ◀

A slightly more involved proof, again by using shifting arguments that exploit the finiteness of the worst-case scenario set, gives the following lemma.

▶ **Lemma 13.** *Let A be an optimal appointment schedule. Then for all jobs i, $1 \leq i \leq n-1$, there is a worst-case scenario $P \in \mathcal{W}_A$ such that job i is not overaged and job $i+1$ is not underaged.*

Proof. Consider an arbitrary, but fixed job i, $1 \leq i \leq n-1$. By Lemma 12, there is at least one scenario in $\mathcal{W}_A$ in which i is not overaged. Suppose that all scenarios in $\mathcal{W}_A$ in which job i is not overaged, also have job $i+1$ underaged. Let c^* be the worst-case scenario cost of A. For a given $\epsilon > 0$, let A^ϵ be the appointment schedule in which $A^\epsilon_{i+1} = A_{i+1} + \epsilon$, and $A^\epsilon_j = A_j$ for all other jobs j. As $\mathcal{P}$ is finite, and as $F(A, P)$ is a continuous function in A, there exists $\epsilon > 0$ such that the schedule A^ϵ satisfies the following properties:

- For all $P \in \mathcal{W}_A$, $F(A^\epsilon, P) \leq F(A, P)$.
- For all $P \in \mathcal{W}_A$ such that job i is overaged in schedule A, $F(A^\epsilon, P) < F(A, P)$.
- For all $P \in \mathcal{P} \setminus \mathcal{W}_A$, $F(A^\epsilon, P) < c^*$. In particular, the worst-case scenarios for the appointment schedule A^ϵ belong to the set $\mathcal{W}_A$.
- For the schedule A^ϵ, in no scenario from the set $\mathcal{P}$ does job i finish exactly at time A^ϵ_{i+1}.

Again, as $\mathcal{P}$ is finite and as the cost function is continuous with respect to the appointment schedule, there is $0 < \delta < \epsilon$ such that decreasing a^ϵ_i to $a^\epsilon_i - \delta$ lets no scenario in $\mathcal{P} \setminus \mathcal{W}_{A^\epsilon}$ reach c^*, but all scenarios in which job i is underaged for schedule A have a lower cost as compared to A^ϵ. Thus, the worst-case scenario cost of the appointment schedule $(a^\epsilon_1, \ldots, a^\epsilon_{i-1}, a^\epsilon_i - \delta, a^\epsilon_{i+1}, \ldots, a^\epsilon_n)$ is less than c^*, contradicting the optimality of the appointment schedule A. ◀

The next lemma establishes a crucial property of the worst-case scenarios of an optimal schedule: any overaged job is followed by an overaged job.

▶ **Lemma 14.** *For an optimal solution A, for every $P \in \mathcal{W}_A$, if job i, for some $1 \leq i \leq n-1$, is overaged, then so is job $i+1$.*

Proof. Assume to the contrary, for some i there is $P_1 \in \mathcal{W}_A$ with job i overaged but job $i+1$ underaged or tight. By Lemma 13, there is a worst-case scenario $P_2 \in \mathcal{W}_A$ with job i underaged or tight, and job $i+1$ tight or overaged. Split the cost of P_1 into $c(P_1) = c_{\leq i}(P_1) + c_{\geq i+2}(P_1) + c_{i+1}(P_1)$, i. e., the cost of jobs up to and including job i, the cost of jobs $i+2$ or later, and the underage cost of job $i+1$. In case $i = n-1$, we set the term $c_{\geq i+2}(P_1)$ equal to zero. For P_2 we split $c(P_2) = c_{\leq i}(P_2) + c_{\geq i+1}(P_2)$. As both are worst-case scenarios we have $c(P_1) = c(P_2)$.

Claim: $c_{\leq i}(P_2) < c_{\leq i}(P_1)$. Else, replace the jobs up to and including i in P_1 by the corresponding jobs in P_2. This *strictly* increases the underage cost of $i+1$, because i is

overaged in P_1, and keeps $c_{\geq i+2}(P_1)$ as it is. This way we get a scenario with strictly higher cost, contradicting $P_1 \in \mathcal{W}_A$.

Because of the claim and $c(P_1) = c(P_2)$ we must have $c_{\geq i+1}(P_2) > c_{\geq i}(P_1) + c_{i+1}(P_1)$. Now, construct a scenario P' using p_j of P_1 for all jobs $1 \leq j \leq i$ and, in case $i \leq n-2$, using p_j of P_2 for all jobs j with $i+2 \leq j \leq n$. As $i+1$ must be shorter in P_1 than in P_2, we can adjust p_{i+1} such that its completion time is the same as in scenario P_2. Thereby, in case $i \leq i-2$, the realized starting time of $i+2$ in P' is that of $i+2$ in scenario P_2.

Now, P' has strictly higher cost than the worst-case scenarios P_1 and P_2, which contradicts the assumption that P_1 and P_2 are worst-case scenarios for the optimal schedule A. Hence the statement of the lemma holds. ◀

The following lemma is the overage analog of Lemma 11.

▶ **Lemma 15.** *If for some optimal solution A and some worst-case scenario $P \in \mathcal{W}_A$ job i is overaged, then $p_i = \overline{p}_i$.*

Proof. If i is overaged, all jobs $j \geq i$ are overaged as well, by Lemma 14. But then, if one could increase the length of job i, the cost of A would increase, which is not possible, because P is a worst-case scenario. Hence $p_i = \overline{p}_i$. ◀

The following lemma shows that many extremal scenarios are simultaneously worst case.

▶ **Lemma 16.** *There exists an optimal appointment schedule for which at least $n+1$ extremal scenarios are worst-case scenarios.*

Proof. Let A^* be an optimal solution of a given instance of the appointment scheduling problem. For each $P \in \mathcal{P}$, the cost of A^* in the scenario P, $F(A^*, P)$ is given by a linear function $f_P(A)$. Here A corresponds to the assigned duration to each job. Consider the following linear program:

$$\begin{aligned} \min \quad & C \\ \text{s.t.} \quad & f_P(A) \leq C, \text{ for all } P \in \mathcal{P}. \end{aligned}$$

Clearly, A^* is a feasible solution of this linear program. By Lemma 7, an optimal solution of this linear program yields an optimal solution of the appointment scheduling problem. The linear program has $n+1$ variables, therefore in an optimal basic feasible solution, at least $n+1$ constraints must be satisfied with equality. Hence there exists an optimal appointment schedule in which at least $n+1$ extremal scenarios are worst-case scenarios. ◀

The next theorem characterizes the set of extremal worst-case scenarios for an optimal appointment schedule.

▶ **Theorem 17.** *There exists an optimal appointment schedule A for which the set of extremal worst-case scenarios is given by*

$$\mathcal{W}_A = \{(\underline{p}_1, \ldots, \underline{p}_i, \overline{p}_{i+1}, \ldots, \overline{p}_n) \ : \ 1 \leq i < n\} \cup \{(\underline{p}_1, \ldots, \underline{p}_n), (\overline{p}_1, \ldots, \overline{p}_n)\}.$$

Proof. Follows immediately from Lemmas 14, 15 and 16. ◀

We are now ready to prove the main theorem of this section.

Proof of Theorem 5. Equating the cost of the worst-case scenarios in which $k-1$ respectively k is the last job at minimal length gives $(a_k - \underline{p}_k)u = (\overline{p}_k - a_k)\sum_{i=k}^{n} o_{\geq i}$ for all $1 < k \leq n$. Equating the scenario with all jobs maximal with the one where only the first is minimal, gives the equation also for $k = 1$. These n equations uniquely determine the optimal schedule as the *balanced schedule*, A^B. Direct calculation of the cost of A^B for any of these scenarios gives the result. ◀

4 Robust Appointment Scheduling and Ordering

Theorem 5 reduces the robust appointment scheduling and ordering problem to finding a permutation $\pi : [n] \to [n]$ that minimizes the cost $F(\pi)$ of the corresponding balanced schedule:

$$\min_{\pi} \sum_{i=1}^{n} \frac{u \sum_{\pi^{-1}(j) \geq \pi^{-1}(i)} o_{\pi^{-1}(j)} \Delta_{\pi^{-1}(i)}}{u + \sum_{\pi(j) \geq \pi^{-1}(i)} o_{\pi^{-1}(j)}} \ .$$

By scaling cost coefficients, we may assume $u = 1$. Using variables $\Theta_i := \Theta_i^{\pi} := \sum_{\pi^{-1}(j) \geq \pi^{-1}(i)} o_{\pi^{-1}(j)}$ and taking cues from single-machine scheduling with weighted-sum-of-completion-times objective [16], we can restate the ordering problem as that of minimizing a concave function over a supermodular polyhedron:

$$\min \quad \sum_{i=1}^{n} \frac{\Delta_i \Theta_i}{1 + \Theta_i} \tag{1a}$$

$$\text{s.t.} \quad \sum_{j \in S} o_j \Theta_j \geq \frac{1}{2}\Big(\sum_{j \in S} o_j\Big)^2 + \frac{1}{2}\sum_{j \in S} o_j^2 \quad \text{for all } S \subseteq N. \tag{1b}$$

Here we use N to denote the set $[n] = \{1, 2, \ldots, n\}$ of all jobs.

▶ **Lemma 18.** *Linear program* 1 *is an exact formulation of the ordering problem.*

Proof. The objective function (1a) is strictly concave, which implies that an optimal solution is attained at an extreme point of the polyhedron described by the inequalities (1b). This polyhedron is supermodular (i. e., the right-hand side of (1b) viewed as a function of $S \subseteq N$ is supermodular). In particular, all extreme points are of the form $(o_{\geq \sigma(1)}, \ldots, o_{\geq \sigma(n)})$, where $\sigma : N \to N$ is a permutation of jobs and $o_{\geq \sigma(i)} = \sum_{j \geq i} o_{\sigma(j)}$ (see, e. g., [7]). This means that an optimal ordering of the jobs can be found by solving for an optimal extreme point of this formulation. Furthermore, if $(\Theta_1^*, \ldots, \Theta_n^*)$ is an optimal extreme point, then an optimal ordering of the jobs is given by a permutation σ which satisfies $\Theta_{\sigma(1)}^* \geq \ldots \geq \Theta_{\sigma(n)}^*$. ◀

The first result of this section is an ad hoc $(2+\epsilon)$-approximation algorithm for the robust appointment scheduling and ordering problem. Observe that for $\Theta_i \leq 1$ we have $\Theta_i \leq 2\frac{\Theta_i}{1+\Theta_i}$, whereas for $\Theta_i \geq 1$ we have $1 \leq 2\frac{\Theta_i}{1+\Theta_i}$. In particular, this readily implies that if o_i happens to be larger than 1 for all jobs i, then so is Θ_i for any ordering, and any ordering is a 2-approximation. If, on the other hand, $\sum_{i=1}^{n} o_i \leq 1$, it is straightforward to show that ordering by Smith's rule [18], i.e, by non-decreasing ratios of $\frac{\Delta_i}{o_i}$, gives a 2-approximation. We will have more to say about this rule later. We first show that the following optimization problem, inspired by these two extremal cases, leads to a $(2+\epsilon)$-approximation in general:

$$\min_{\substack{S \subseteq N \\ \sum_{i \in S} o_i \leq 1}} \Big(\sum_{i \notin S} \Delta_i + \min_{\pi : S \to S} \sum_{i \in S} \Delta_i \Theta_i^{\pi} \Big) \tag{2}$$

Note that after choosing S in Problem (2), Smith's rule on the elements of S yields an optimal order Θ for the "inner" minimization problem. Moreover, even though we do not require $\sum_{i\in S} o_i + o_j > 1$ for all j not in S, any j for which this is not the case can be added to S without increasing the objective function value. Therefore, we assume, for optimal or approximate solutions, that S is maximal with respect to $\sum_{i\in S} o_i \leq 1$.

A solution S of Problem (2) can be turned into an ordering π_S of all jobs in N as follows:

1. Schedule first the jobs not in S in arbitrary order.
2. Afterwards, schedule the jobs in S using Smith's rule.

We first prove a lemma that establishes the connection between an approximate solution to Problem (2) and an optimal solution of the ordering problem.

▶ **Lemma 19.** *Let S^α be an α-approximate solution for Problem (2), and let π_{S^α} be the sequence of jobs produced by the above algorithm, if called with input S^α. Then π_{S^α} is a 2α-approximate solution of the robust appointment scheduling and ordering problem.*

Proof. Let S' be an optimal solution to Problem (2), and let π^* be an optimal ordering. We construct a solution S^* for Problem (2) from π^* as the set of all jobs i for which $\Theta^*_i \leq 1$.

$$
\begin{aligned}
F(\pi_{S^\alpha}) &= \sum_{i\in N} \frac{\Theta^\alpha_i \Delta_i}{1+\Theta^\alpha_i} = \\
\sum_{i\notin S^\alpha} \frac{\Theta^\alpha_i \Delta_i}{1+\Theta^\alpha_i} + \sum_{i\in S^\alpha} \frac{\Theta^\alpha_i \Delta_i}{1+\Theta^\alpha_i} &\leq \sum_{i\notin S^\alpha} \Delta_i + \sum_{i\in S^\alpha} \Delta_i \Theta^\alpha_i \leq \qquad (3)\\
\alpha\left(\sum_{i\notin S'} \Delta_i + \sum_{i\in S'} \Delta_i \Theta'_i\right) &\leq \alpha\left(\sum_{i\notin S^*} \Delta_i + \sum_{i\in S^*} \Delta_i \Theta^*_i\right) \leq \qquad (4)\\
2\alpha\left(\sum_{i\notin S^*} \frac{\Theta^*_i \Delta_i}{1+\Theta^*_i} + \sum_{i\in S^*} \frac{\Theta^*_i \Delta_i}{1+\Theta^*_i}\right) &= 2\alpha \sum_{i\in N} \frac{\Theta^*_i \Delta_i}{1+\Theta^*_i} = 2\alpha F(\pi^*).
\end{aligned}
$$

Inequality (3) is trivial as $\Theta^\alpha_i \geq 0$. The inequality in changing to line (4) is given by the approximation factor provided by the antecedent of the lemma. The inequality in line (4) holds by the optimality of S'. The last inequality follows from the construction of S^* because we have $1 \leq 2\frac{\Theta^*_i}{1+\Theta^*_i}$ for all i not in S^* and $\Theta^*_i \leq 2\frac{\Theta^*_i}{1+\Theta^*_i}$ for all i in S^*. ◀

From Lemma 19, it follows that in order to obtain a 2α-approximation algorithm for the ordering problem, it suffices to obtain an α-approximation algorithm for Problem (2). We show that we can actually obtain an FPTAS for this optimization problem by re-casting it as a modified knapsack problem.

Without loss of generality we assume that the jobs are sorted in the order of Smith's rule, i. e., $\Delta_1/o_1 \leq \ldots \leq \Delta_n/o_n$. Assuming rational input we can find integers Q and q_i such that $o_i = q_i/Q$ for all i. Since the objective function in Problem (2) is linear, it suffices to consider the optimization problem in which the objective function is scaled by Q. Problem (2) can then be recast as follows:

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} \Delta_i \sum_{j=i}^{n} q_j x_j + \sum_{i=1}^{n} Q\Delta_i (1-x_i) && (5)\\
\text{s. t.} \quad & \sum_{i=1}^{n} q_i x_i \leq Q, && (6)\\
& x_i \in \{0,1\} \text{ for all } i = 1,\ldots,n. && (7)
\end{aligned}
$$

Let x' be an optimal solution to the modified knapsack problem given by (5)-(7). Then an optimal solution S' of Problem (2) can be obtained as $S' = \{i \in N : x'_i = 1\}$. This problem can be solved in pseudo-polynomial time using dynamic programming, as stated in the following lemma.

▶ **Lemma 20.** *The modified knapsack problem given by* (5)-(7) *can be solved in pseudo-polynomial time by dynamic programming.*

Proof. We reduce the problem of solving the modified knapsack problem to that of the shortest path problem in a graph. For a problem instance corresponding to (5)-(7), we construct the corresponding graph G as follows. The nodes in the graph are indexed by (i, v) for $1 \leq i \leq n+1$ and $0 \leq v \leq Q$, where i corresponds to a job and v corresponds to the total contribution to the overage cost based on whether we decide to select the jobs for set S or not. From each node (i, v), there are at most two outgoing arcs:

1. Arc to node $(i-1, v+q_i)$ of cost $\Delta_i(v+q_i)$, provided $v+q_i \leq Q$ (this corresponds to choosing job i for set S).
2. Arc to node $(i-1, v)$ of cost $\Delta_i Q$ (this corresponds to not choosing job i for the set S).

All the nodes of the form $(1, v)$ in the graph are connected to a terminal node T. The optimal solution of the modified problem corresponds to a shortest path in graph G from the node $(n+1, 0)$ to the node T. Since there are $\mathrm{O}(nQ)$ nodes and arcs in the graph, the shortest path problem (and hence the modified knapsack problem) can be solved in the same amount of time. ◀

Similar to the FPTAS for the knapsack problem [11], we can obtain an FPTAS for the modified knapsack problem.

▶ **Lemma 21.** *There exists an FPTAS for the modified knapsack problem given by* (5)-(7).

Proof. The objective function (5) can be written as $g(x) = \sum_{i=1}^{n} a_i x_i + \sum_{i=1}^{n} b_i(1-x_i)$, where $a_i = \Delta_i \sum_{j=1}^{i} q_j$ and $b_i = \Delta_i Q$ for $i = 1, \ldots, n$. Let $C = \max_i(a_i, b_i)$. Consider the modified objective function given by

$$g'(x) = \sum_{i=1}^{n} a'_i x_i + \sum_{i=1}^{n} b'_i(1-x_i), \tag{8}$$

where $a'_i = \lfloor a_i n/\epsilon C \rfloor$ and $b'_i = \lfloor b_i n/\epsilon C \rfloor$. In $g'(x)$, all the coefficients have value at most n/ϵ. Using the dynamic programming algorithm given in the proof of Lemma 20, the problem with the modified objective function can be solved in $O((n^2/\epsilon))$ time, which is polynomial in n and $1/\epsilon$. It remains to show that the optimal solution for the problem with objective function $g'(x)$ is a $(1+\epsilon)$-approximation to the optimal solution of the problem with the original objective function g.

Let S^* (resp. S') correspond to the set of all jobs for which the corresponding variable x_i is 1 in the optimal solution for the objective function $g(x)$ (resp. $g'(x)$). Then, an upper

bound on the objective function value of the solution S' is given by

$$\begin{aligned} g(S') &= \sum_{i\in S'} a_i + \sum_{i\notin S'} b_i \\ &= \frac{\epsilon C}{n}\sum_{i\in S'}\frac{a_i n}{\epsilon C} + \frac{\epsilon C}{n}\sum_{i\notin S'}\frac{b_i n}{\epsilon C} \\ &\leq \frac{\epsilon C}{n}\sum_{i\in S'}\left(\left\lfloor\frac{a_i n}{\epsilon C}\right\rfloor + 1\right) + \frac{\epsilon C}{n}\sum_{i\notin S'}\left(\left\lfloor\frac{b_i n}{\epsilon C}\right\rfloor + 1\right) \\ &\leq \frac{\epsilon C}{n} g'(S') + \epsilon C \\ &\leq \frac{\epsilon C}{n} g'(S^*) + \epsilon C, \end{aligned}$$

where the last inequality follows from the fact that S' is an optimal solution for the objective function g'. Further, a lower bound on the objective function value of the optimal solution S^* is as follows:

$$\begin{aligned} g(S^*) &= \sum_{i\in S^*} a_i + \sum_{i\notin S^*} b_i \\ &\geq \frac{\epsilon C}{n}\left(\sum_{i\in S^*}\left\lfloor\frac{a_i n}{\epsilon C}\right\rfloor + \sum_{i\notin S^*}\left\lfloor\frac{b_i n}{\epsilon C}\right\rfloor\right) \\ &= \frac{\epsilon C}{n} g'(S^*). \end{aligned}$$

Putting both inequalities together, it follows that $g(S') \leq g(S^*) + \epsilon C \leq (1+\epsilon) g(S^*)$, as $g(S^*) \geq C$. Therefore, S' is a $(1+\epsilon)$-approximate solution to the modified knapsack problem. ◀

Putting the pieces together gives the first constant-factor approximation algorithm for the ordering problem.

▶ **Theorem 22.** *There is a* $(2+\epsilon)$*-approximation algorithm for the* robust appointment scheduling and ordering problem.

Proof. Follows from Lemma 19 and Lemma 21. ◀

An advantage of the $(2+\epsilon)$-approximation algorithm, apart from being ad hoc and using problem-specific insights, is that the jobs in the "non-Smith" part of the sequence can be scheduled in arbitrary order, which is especially appealing from a practical point of view. Interestingly, Lemma 18 actually opens the door to the use of more general machinery developed for the family of single-machine scheduling problems $1|\,|\sum w_j f(C_j)$. Indeed, Lemma 18 implies that the robust appointment scheduling and ordering problem is equivalent to a single-machine scheduling problem of this type, where f is increasing and concave. For arbitrary concave f, it was shown in [19] that Smith's rule yields an approximation guarantee of $(\sqrt{3}+1)/2$. With the help of the theory developed in [10], one can actually get a refined approximation factor of 1.137 for Smith's rule for the particular concave function encountered here. Moreover, it follows from recent work (see [19, 14]) on $1|\,|\sum w_j f(C_j)$ with concave resp. non-decreasing functions f that there is in fact a PTAS for the robust appointment scheduling and ordering problem.

We leave the computational complexity of the robust appointment scheduling and ordering problem as an open problem, and note that it is also unknown whether minimizing (monotone) concave functions over supermodular polyhedra is NP-hard.

Acknowledgments. The authors thank Mehmet Begen, Diwakar Gupta, Jim Orlin, Larry Robinson, David Shmoys, and Jose Verschae for stimulating discussions.

References

1 Mehmet A. Begen, Retsef Levi, and M. Queyranne. A sampling-based approach to appointment scheduling. Technical report, Sauder School of Business, University of British Columbia, 2008. Working Paper.

2 Mehmet A. Begen and Maurice Queyranne. Appointment scheduling with discrete random durations. *Mathematics of Operations Research*, 36:240–257, 2011.

3 Aharon Ben-Tal and Arkadi Nemirovski. Robust optimization - methodology and applications. *Mathematical Programming*, 92:453–480, 2002.

4 Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Operations Research*, 52:35–53, 2004.

5 T. Cayirli and E. Veral. Outpatient scheduling in healthcare: a review of literature. *Production and Operations Management*, 12:519–549, 2003.

6 Brian Denton and Diwakar Gupta. A sequential bounding approach for optimal appointment scheduling. *IIE Transactions*, 35:1003–1016, 2003.

7 Satoru Fujishige. *Submodular Functions and Optimization*, volume 58 of *Annals of Discrete Mathematics*. Elsevier, 2005. 2nd edition.

8 Linda V. Green, Sergei Savin, and Ben Wang. Managing patient service in a diagnostic medical facility. *Operations Research*, 54:11–25, 2006.

9 Diwakar Gupta. Surgical suites' operations management. *Productions and Operations Management*, 16:689–700, 2007.

10 Wiebke Höhn and Tobias Jacobs. On the performance of Smith's rule in single-machine scheduling with nonlinear cost. In *LATIN*, pages 482–493, 2012.

11 Oscar H. Ibarra and Chul E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM*, 22:463–468, 1975.

12 Satoru Iwata. Submodular function minimization. *Mathematical Programming*, 112:45–64, 2008.

13 Guido C. Kandoorp and Ger Koole. Optimal outpatient appointment scheduling. *Health Care Management Science*, 10:217–229, 2007.

14 Nicole Megow and José Verschae. Dual techniques for scheduling on a machine with varying speed. In *ICALP*, pages 745–756, 2013.

15 James B. Orlin. A faster strongly polynomial time algorithm for submodular function minimization. *Mathematical Programming*, 118:237–251, 2009.

16 Maurice Queyranne. Structure of a simple scheduling polyhedron. *Mathematical Programming*, 58:263–285, 1993.

17 Lawrence W. Robinson and Rachel R. Chen. Scheduling doctor's appointments: Optimal and empirically-based heuristic policies. *IIE Transactions*, 35:295–307, 2003.

18 W. E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3:59–66, 1956.

19 Sebastian Stiller and Andreas Wiese. Increasing speed scheduling and flow scheduling. In *ISAAC*, pages 279–290, 2010.

20 P. Patrick Wang. Static and dynamic scheduling of customer arrivals to a single-server system. *Naval Research Logistics*, 40:345–360, 1993.

21 P. Patrick Wang. Sequencing and scheduling n customers for a stochastic server. *European Journal of Operational Research*, 119:729–738, 1999.

Computational Complexity of Certifying Restricted Isometry Property

Abhiram Natarajan and Yi Wu

Purdue University
West Lafayette, IN, USA
{nataraj2,wu510}@cs.purdue.edu

Abstract

Given a matrix A with n rows, a number $k < n$, and $0 < \delta < 1$, A is (k, δ)-RIP (Restricted Isometry Property) if, for any vector $x \in \mathbb{R}^n$, with at most k non-zero co-ordinates,

$$(1-\delta)\|x\|_2 \leq \|Ax\|_2 \leq (1+\delta)\|x\|_2$$

In other words, a matrix A is (k, δ)-RIP if Ax preserves the length of x when x is a k-sparse vector. In many applications, such as compressed sensing and sparse recovery, it is desirable to construct RIP matrices with a large k and a small δ. It is known that, with high probability, random constructions produce matrices that exhibit RIP. This motivates the problem of certifying whether a randomly generated matrix exhibits RIP with suitable parameters.

In this paper, we prove that it is hard to approximate the RIP parameters of a matrix assuming the Small-Set-Expansion Hypothesis. Specifically, we prove that for any arbitrarily large constant $C > 0$ and any arbitrarily small constant $0 < \delta < 1$, there exists some k such that given a matrix M, it is Small-Set-Expansion-hard to distinguish the following two cases:

- (Highly RIP) M is (k, δ)-RIP.
- (Far away from RIP) M is not $(k/C, 1-\delta)$-RIP.

Most of the previous results on the topic of hardness of RIP certification only hold for certification when $\delta = o(1)$; i.e, when the matrix exhibits strong RIP. In practice, it is of interest to understand the complexity of certifying a matrix with δ being close to $\sqrt{2}-1$, as it suffices for many real applications to have matrices with $\delta = \sqrt{2}-1$. Our hardness result holds for any constant δ. Specifically, our result proves that even if δ is indeed very small, i.e. the matrix is in fact *strongly RIP*, certifying that the matrix exhibits *weak RIP* itself is Small-Set-Expansion-hard.

In order to prove the hardness result, we prove a variant of the Cheeger's Inequality for sparse vectors. Although a similar result is already known, our proof technique gives better constants in the inequality which may be useful for other applications. Specifically, let A be the adjacency matrix of a d-regular graph $G(V, E)$, and $L = I - \frac{1}{d}A$ be the normalized Laplacian matrix of G. For any $\eta \leq 1/2$, we show that

$$\lambda_\eta \leq \phi_\eta(G) \leq \sqrt{(2-\lambda_\eta)\lambda_\eta}$$

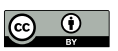
where $\lambda_\eta = \min_{\|x\|_0 = \eta n} \frac{x^T L x}{|x|_2^2}$ and $\phi_\eta(G)$ is the minimum edge expansion among all the sets of size at most $\eta|V|$.

It is interesting to note that the relationship between λ_η and $\Phi_\eta(G)$ is different from (and tighter than) the relation between λ and $\phi(G)$ in the regular version of Cheeger's Inequality (which states that $\frac{\lambda}{2} \leq \phi(G) \leq \sqrt{2\lambda}$). We will see that obtaining this tighter relationship between λ_η and $\phi_\eta(G)$ is crucial in proving our hardness result.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Restricted Isometry Property, RIP, RIP Certification, Sparse Cheeger Inequality, SSE Hard

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.371



17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 371–380

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

1 Introduction

Moore's law has enabled the creation of very robust and effective sensing systems. As a result of the ubiquity of such systems, the amount of data generated by these systems has increased vastly. In fact, in most real applications, there is so much data that sampling at the required rates (called Nyquist rate) becomes impractical due to data storage problems as well as the sheer magnitude of the sampling rate [8]. Signal processing literature shows us that this problem is circumvented by constructing compressible representations of signals. This technique leverages the notion of *sparse approximation* and is called *compressed sensing*.

A formal statement of the central problem of compressed sensing is as follows. Assume the presence of a matrix $\Phi \in \mathbb{R}^{m \times n}$, called the sensing matrix, with $m \ll n$. We are also given a vector $\mathrm{y} \in \mathbb{R}^m$, which contains a set of m measurements. We are interested in reconstructing $x \in \mathbb{R}^n$, such that

$$\mathrm{y} = \Phi x$$

Given that $m \ll n$, this setting is under-determined. However, under the completely reasonable premise that x is compressible, i.e., it is well approximated by k-sparse representations, the problem of recovering x becomes feasible. In other words, if we restrict ourselves to vectors which have at most k non zero co-efficients, i.e. $\|x\|_0 = k$ and $k \ll n$, we can efficiently search for solutions. In fact, Candes et al. [5,6,7] show that it is possible recover a k-sparse x *exactly* if the sensing matrix Φ exhibits the *Restricted Isometry Property* (RIP).

▶ **Definition 1.** A matrix Φ is said to exhibit (k, δ)-RIP iff $\forall x \in \mathbb{R}^n$ with $\|x\|_0 = k$

$$(1 - \delta)\|x\|_2 \leq \|\Phi x\|_2 \leq (1 + \delta)\|x\|_2$$

Please note that k might be referred to as 'order' and δ might be referred to as 'Restricted Isometry Constant (RIC)'. Candes et al. showed that it is possible to reconstruct a k-sparse x very efficiently, by solving the minimization problem

$$\min_{\mathrm{a} \in \mathbb{R}^n} \|\mathrm{a}\|_1 \qquad \text{subject to} \quad \Phi \mathrm{a} = \mathrm{y}$$

when we have RIP matrices[1] with $\delta < \sqrt{2} - 1$. Their work has resulted in significant effort towards both deterministic and probabilistic constructions of RIP matrices. Naturally, we want to be able to construct RIP matrices with k as large as possible, but deterministic constructions, such as those presented by Bourgain et al. [4] and DeVore [9], cannot produce RIP matrices of order much greater than $\sqrt{n}$. Deterministic constructions are far from achieving the orders achieved by probabilistic constructions. On the other hand, it has been proven that ± 1 symmetric Bernoulli matrices, or matrices formed by sampling from a Gaussian distribution $\mathcal{N}(0, \frac{1}{n})$, satisfy RIP with $k \in \Theta(n)$ [3,21] with high probability.

The superiority of random constructions motivates the problem of certifying whether a randomly drawn matrix Φ, from any of the models mentioned above, exhibits RIP with the required parameters. If we find that a randomly drawn matrix is unsuitable to our purposes, we re-generate it and repeat the certification process. Terry Tao posted the following question on his blog [18]: "An alternate approach (to deterministic construction of RIP matrix), and one of interest in its own right, is to work on improving the time it takes to verify that a

[1] Henceforth, we shall use the phrase 'RIP matrices' instead of saying 'matrices exhibiting the RIP' every time. It is worth noting that Tao once used the abbreviation 'UUP', which stands for 'Uniform Uncertainty Principle', for what is now commonly known as RIP.

given matrix (possibly one of a special form) obeys the UUP (RIP)." In this paper, we prove that RIP certification is NP-hard to approximate in a *strong* sense assuming the truth of the Small-Set-Expansion Hypothesis.

We now state Small-Set-Expansion Hypothesis, which was proposed by Raghavendra and Steurer [14], and is one of the most important conjectures in complexity theory. In order to present the conjecture, we first define the expansion of a graph.

▶ **Definition 2.** Given a graph d-regular graph $G(V, E)$ with n vertices, we define the expansion of a non-empty set $S \subseteq V$ as

$$\phi_G(S) = \frac{|E(S, V-S)|}{d \cdot \min(|S|, |V-S|)}.$$

where $E(S, V-S)$ denotes the collection of edges of G that have one end in S and the other end in $V-S$. The expansion of the graph G is defined as the minimum expansion among all subset of its vertices:

$$\phi(G) = \min_{S \subseteq V} \phi(S).$$

For any $0 < \delta \leq 1/2$, we also define the minimum expansion among all subsets of size $\leq \delta n$ as

$$\phi_\delta(G) = \min_{\substack{S \subseteq V \\ S \leq \delta n}} \phi(S).$$

The Small Set Expansion conjecture states that:

▶ **Conjecture 3.** *For every $\epsilon > 0$, $\exists\, 0 \leq \delta \leq \frac{1}{2}$, such that it is NP-hard to distinguish between:*

- *$\exists\, S \subseteq V$, with $|S| = \delta|V|$, such that $\phi_G(S) \leq \epsilon$*
- *$\forall S \subseteq V$, with $|S| \leq \delta|V|$, we have $\phi_G(S) \geq 1 - \epsilon$*

1.1 Our Main Result

We give a gap preserving reduction from the Small-Set-Expansion problem to the RIP certification problem. More formally, we prove the following theorem:

▶ **Theorem 4.** *For any $0 \leq \delta \leq 1$, and $C \geq 1$, there exists k such that, given a matrix M it is* Small-Set-Expansion-hard[2] *to distinguish between:*

- *(Highly RIP) M is (k, δ)-RIP.*
- *(Far away from RIP) M is not $(k/C, 1-\delta)$-RIP.*

We claim that our result has a very strong form, which we will justify in more detail a little later. Also, as corollaries, we have that

▶ **Corollary 5.** *Given a matrix M and k, it is* Small-Set-Expansion-hard *to distinguish whether the matrix is (k, δ)-RIP or not $(k, 1-\delta)$-RIP for any $\delta > 0$*

▶ **Corollary 6.** *Given a fixed δ and matrix M, it is* Small-Set-Expansion-hard *to get a constant approximation for the smallest k such that M exhibits (k, δ)-RIP.*

[2] A problem $\mathcal{I}$ is defined to be Small-Set-Expansion-hard if Small-Set-Expansion $\leq_P \mathcal{I}$

1.2 Comparison with Previous Work

Let us go over some previous work on the topic of hardness of RIP certification, and also make a few observations about Theorem 4 to justify our claim that we are proving hardness of RIP certification in a very strong sense. Bandeira et al. [2] prove that the exact decision version of the problem of RIP certification is NP-hard. In other words, they proved that given δ, k, it is NP-hard to certify whether a matrix exhibits (k, δ)-RIP or not. It was later established by Tillmann and Pfetsch [19] that the same problem is also co-NP-hard. Both works reduce from the problem of determining the spark of a matrix, which is known to be NP-hard. It should be mentioned that δ in both results is in $o_n(1)$, where n is the number of rows of the matrix. Also, we must note that the exact decision version of the problem is restrictive.

Results by Koiran and Zouzias (KZ) are the only works we know of on the approximation version of the problem. KZ obtain various inapproximability results by making assumptions on the hidden clique problem [12] and the dense subgraph problem [11]. Most of the results are of the form that, for some k, δ_1, δ_2, (depending on the assumption used), it is hard to distinguish whether a matrix is (k, δ_1)-RIP, or not (k, δ_2)-RIP. In almost all of the cases, δ_1, δ_2 are $\in o_n(1)$, with the exception of one result, which we shall state below:

- No polynomial time algorithm can distinguish matrices that satisfy the $(\Theta(n), \frac{\kappa}{2})$-RIP from matrices that do not satisfy the $(\Theta(n), \kappa)$-RIP

where $\kappa \leq \frac{\sqrt{5}}{3}$ is an unknown constant depending on the correctness of hidden-clique and densest-subgraph conjectures. In practice, it is known that an RIP matrix is useful for many applications as long as $\delta \leq \sqrt{2} - 1$. Clearly, the above theorem does not rule out the existence of an algorithm for deciding whether the RIC of a matrix is $\leq \sqrt{2} - 1$. This is because there is no guarantee that $\kappa \in (\sqrt{2} - 1, 2\sqrt{2} - 2)$. KZ also state "...our hardness results do not rule out the existence of a polynomial-time algorithm distinguishing between matrices with a very small RIP parameter and matrices with a RIP parameter larger than say 0.1...".

Theorem 4 is clearly equipped to make stronger statements than any previous work on inapproximability of RIP parameters. It suggests that certifying RIP for any constant $0 < \delta < 1$ is Small-Set-Expansion-hard. In addition, even if the matrix indeed exhibits *strong* RIP (small constant δ and very large k), it is still Small-Set-Expansion-hard to even certify if it exhibits *weak RIP* (with large δ close 1 and small k).

1.3 Proof Overview

Let us assume that G is a d-regular graph with adjacency matrix A, and $L = I - \frac{1}{d}A$ is the normalized Laplacian matrix of the graph. it is easy to see that, given $x_S \in \{0, 1\}^n$ as the indicator vector of set S, we have that

$$\phi(S) = \frac{x_S^T L x_S}{{x_S}^{\mathrm{T}} x_S} = \frac{\|M x_S\|_2^2}{\|x_S\|_2^2}$$

for M satisfying $M^{\mathrm{T}} M = L$. We know that the Laplacian is a quadratic form and thus is positive semi-definite. Thus, L always admits the decomposition $L = M^{\mathrm{T}} M$.

The strategy of the reduction is to take a Small-Set-Expansion instance and to construct the corresponding M for the RIP certification problem. Our reduction has a similar flavor to the reduction in Koiran and Zouzias [12] (they call their reduction as *Cholesky Reduction*). If there is a small set S with expansion less than ϵ, we know that x_S,

corresponding to this S, is a sparse vector such that $\|Mx_S\|_2 \leq \sqrt{\epsilon}\|x_S\|_2$ and this suggests that M is far from being a RIP matrix. The second case of the proof is more involved. Here, we would like to show that if there exists a k-sparse $x \in \mathbb{R}^n$ such that $x^{\mathrm{T}}Lx$ is bounded away from 1, then we can find a small set S such that $\phi(S)$ is also bounded away from 1.

If we ignore the sparsity constraint, this kind of conversion from a real vector x to a boolean vector x_S is exactly reminiscent of the "hard direction" of Cheeger's Inequality [1, 13, 16]. In this paper, we prove the following generalization of Cheeger's Inequality for sparse vectors, which we use to prove the "hard direction".

▶ **Theorem 7.** *(Sparse Cheeger's Inequality) Let A be the adjacency matrix of a d-regular graph G, and $L = I - \frac{1}{d}A$ be its normalized Laplacian matrix. For any $\delta \leq 1/2$, we have that*

$$\lambda_\delta \leq \phi_\delta(G) \leq \sqrt{(2-\lambda_\delta)\lambda_\delta}$$

where $\lambda_\delta = \min_{|x|_0 = \delta|V|} \frac{x^{\mathrm{T}}Lx}{x^{\mathrm{T}}x}$.

The above inequality establishes a relationship between the minimum expansion of G on small sets with the minimum value of $\frac{x^{\mathrm{T}}Lx}{x^{\mathrm{T}}x}$ for sparse real vectors x. A similar and independent, but not identical, result is known – Theorem 2.1 in Steurer [17]. We observe that the constants in Theorem 7 are better, and this might find applications elsewhere.

For the purpose of comparison, we also list the original Cheeger's Inequality here:

▶ **Theorem 8.** *(Cheeger's Inequality) Let A be the adjacency matrix of a graph G, and $L = I - \frac{1}{d}A$ be its normalized Laplacian matrix. We have that*

$$\frac{\lambda}{2} \leq \phi(G) \leq \sqrt{2\lambda}$$

where

$$\lambda = \min_{\substack{x \in \mathbb{R}^n \\ Lx \neq \vec{0}}} \frac{\|x^T Lx\|_2}{\|x\|_2^2}$$

is the second smallest eigenvalue of L.

It is interesting to note here that the relationship between λ_δ and $\phi_\delta(G)$ in Theorem 7 is tighter than the relationship between λ and $\phi(G)$ in Theorem 8. It is crucial for our proof that we get $\sqrt{(2-\lambda_\delta)\lambda_\delta}$ instead of $\sqrt{2\lambda_\delta}$ in Theorem 7. We need to prove that if there exists a δ-sparse vector x[3] such that λ_δ is bounded away from 1, then there is a small set whose expansion is also bounded away from 1. If what we had was only $\phi_\delta(G) \leq \sqrt{2\lambda_\delta}$, the bound becomes trivial even when we know that $\lambda_\delta = 1/2$. It is only because of Theorem 7 that we find that, as long as λ_δ is bounded away from 1, we also have that $\sqrt{(2-\lambda_\delta)\lambda_\delta}$ is bounded away from 1. This turns out to be exactly what we need to prove.

The proof of the sparse Cheeger's Inequality bears resemblance to the proof of the classical Cheeger's Inequality (e.g., see [20]). The proof makes it necessary to strengthen the analysis for the sparse vector case so as to obtain a tighter relationship between λ_δ and $\Phi_\delta(G)$.

One final thing to notice is that our hardness result amplifies the dependence on the order k. We show that it is hard to distinguish $(k, 1-\delta)$-RIP from $(k/C, \delta)$-RIP, where C is any arbitrary constant. To this end, we need to use an equivalent statement of the Small-Set-Expansion Hypothesis by Raghavendra et al. [15] which gives us a stronger starting point for the hardness reduction.

[3] $\|x\|_0 \leq \delta n$

1.4 Organization

The paper is organized as follows. In Section 2, we prove the sparse Cheeger's Inequality, namely Theorem 7. We then use this theorem to prove our main result, i.e Theorem 4 which is presented in Section 3.

2 Sparse Cheeger's Inequality

Below we state the proof of Theorem 7.

Proof. Assuming that $|V| = n$, let us first prove the left side of the inequality, analogous to what is commonly called the *easy direction*. Choose $x_S \in \{0,1\}^n$, as a bit vector representation of a set S of size at most $\delta|V|$. We then easily get

$$\phi_\delta(G) = \frac{x_S^T L x_S}{\|x_S\|^2} \leq \lambda_\delta$$

Next, we prove the right hand side of inequality in Theorem 7. Assume we are given any $x \in \mathbb{R}^n$ such that

$$\frac{x^T L x}{\|x\|_2^2} = \lambda_\delta$$

We shall prove that, using x, we will be able to construct some set S such that $\phi(S) \leq \sqrt{(2-\lambda_\delta)\lambda_\delta}$. This will complete the proof because $\phi_\delta(G)$ is the minimum value of $\phi(S)$ over all S with $|S| \leq \delta n$.

Let us use x_i to indicate the i-th coordinate of x, by the property of the Laplacian matrix of a graph, we know that

$$\frac{x^T L x}{\|x\|_2^2} = \frac{\sum_{1\leq i,j\leq n}(x_i - x_j)^2 A_{ij}}{2d\sum_{i=1}^n |x_i|^2} \tag{1}$$

Without loss of generality, we can assume that all coordinates x_i have nonnegative value because changing x_i to $|x_i|$ does not increase $\sum_{1\leq i,j\leq n}(x_i - x_j)^2 A_{ij}$ and $\sum |x_i|^2$ remains the same. Also, we can scale x such that $0 \leq x_1 \leq x_2 \ldots \leq x_n = 1$ because $\frac{x^T L x}{\|x\|_2^2}$ does not change when we scale x.

Consider a distribution P with density $f(x) = 2x$ on the interval $(0,1)$. It is easy to verify this is a valid density function as

$$\int_0^1 2x \cdot \mathrm{d}x = 1.$$

Now consider the following randomized construction of set S from x.

1. Choose t in $(0,1)$ according to P
2. Set S to be $S_t = \{i \mid x_i \geq t\}$

Given the fact that x is δn sparse, $|S_t| \leq \delta n$ for any $0 < t < 1$. We can easily see that for any $0 \leq a \leq b \leq 1$, we have

$$\Pr\left(t \in [a,b]\right) = \int_a^b 2x dx = b^2 - a^2.$$

Therefore,

$$\Pr(x_i \in S_t) = \Pr(x_i \geq t) = x_i^2$$

which implies that

$$\mathbf{E}_t\left[|S_t|\right] = \sum_{i=1}^{n} x_i^2$$

Also we know that for any i, j, there is an edge between vertex i and vertex j only if t falls between x_i and x_j. Therefore,

$$\mathbf{E}_t\left[|E(S_t, V - S_t)|\right] = \frac{1}{2}\sum_{i,j} A_{ij}|x_i^2 - x_j^2| = \frac{1}{2}\sum_{1\le i,j\le n} A_{ij}|x_i - x_j||x_i + x_j|$$
$$\le \sqrt{\frac{\sum_{1\le i,j\le n} A_{ij}(x_i + x_j)^2}{2}} \cdot \sqrt{\frac{\sum_{1\le i,j\le n} A_{ij}(x_i - x_j)^2}{2}}$$

The last inequality in the above sequence of steps is due to the Cauchy-Schwarz Inequality. We then calculate ratio between $\mathbf{E}_t\left[|E(S_t, V - S_t)|\right]$ and $d \cdot \mathbf{E}_t\left[|S_t|\right]$, which is

$$\frac{\mathbf{E}_t\left[|E(S_t, V - S_t)|\right]}{d \cdot \mathbf{E}\left[|S_t|\right]} \le \sqrt{\frac{\sum_{1\le i,j\le n} A_{ij}(x_i + x_j)^2}{2d\sum_{i=1}^n x_i^2}}\sqrt{\frac{\sum_{1\le i,j\le n} A_{ij}(x_i - x_j)^2}{2d\sum_{i=1}^n x_i^2}}.$$

By Equation (1), we know that

$$\frac{\sum_{1\le i,j\le n} A_{ij}(x_i - x_j)^2}{2d\sum_{i=1}^n x_i^2} = \frac{x^T L x}{|x|^2} = \lambda_\delta$$

We also know that

$$\frac{\sum_{1\le i,j\le n} A_{ij}(x_i + x_j)^2}{2d\sum_{i=1}^n x_i^2} + \frac{\sum_{1\le i,j\le n} A_{ij}(x_i - x_j)^2}{2d\sum_{i=1}^n x_i^2} = \frac{\sum_{1\le i,j\le n} A_{ij}(2x_i^2 + 2x_j^2)}{2d\sum_{i=1}^n x_i^2} = 2$$

which implies that

$$\frac{\sum_{1\le i,j\le n} A_{ij}(x_i + x_j)^2}{2d\sum_{i=1}^n x_i^2} = 2 - \lambda_\delta$$

This suggests that

$$\frac{\mathbf{E}_t[|E(S_t, V - S_t)|]}{d \cdot \mathbf{E}_t[|S_t|]} \le \sqrt{\lambda_\delta(2 - \lambda_\delta)}$$

or equivalently

$$\mathbf{E}_t\left[|E(S_t, V - S_t)| - \sqrt{\lambda_\delta(2 - \lambda_\delta)} \cdot d \cdot |S_t|\right] \le 0$$

Therefore, there must exist some $t \in (0, 1)$ such that

$$|E(S_t, V - S_t)| - \sqrt{\lambda_\delta(2 - \lambda_\delta)} \cdot d \cdot |S_t| \le 0$$

or in other words

$$\phi(S_t) = \frac{|E(S, V - S)|}{d \cdot |S_t|} \le \sqrt{\lambda_\delta(2 - \lambda_\delta)}$$

Therefore, if we choose the best $t \in (0, 1)$, we know that

$$\min_t \phi(S_t) \le \sqrt{\lambda_\delta(2 - \lambda_\delta)}$$

This finishes the proof for the right hand side of the inequality in Theorem 7. ◀

3 Proof for the Hardness of Certifying RIP

3.1 Equivalent Variant of the Small-Set-Expansion Hypothesis

The starting point is the following Theorem 9 from [15], which states that a strengthened form of Small-Set-Expansion Hypothesis is equivalent to the original Small-Set-Expansion Hypothesis.

▶ **Theorem 9.** *Given a d-regular graph $G(V, E)$, for all constant integer $q \in \mathbb{N}$, and any constant $\gamma, \epsilon \in (0, 1)$, it is* Small-Set-Expansion-hard *to distinguish the following two cases:*

- *there are q disjoint sets $S_1, S_2, \ldots, S_q \subseteq V$ of size $\frac{n}{q}$ such that $\phi(S) \leq \epsilon + o(\epsilon)$.*
- *for any $0 < \beta < 1$, every set of $S \leq \beta n, S \subseteq V$ has expansion at least $1 - \frac{T_{1-\epsilon}(\beta)}{\beta} - \gamma/\beta$*

Here $T_{1-\epsilon}$ is related to the Gaussian Stability function, which is defined by Khot et al. in [10]. We will use the following upper bound that was presented in [10]:

$$\frac{T_{1-\epsilon/2}(\beta)}{\beta} \leq \beta^{\epsilon/4} \tag{2}$$

for any β, ϵ. By putting $\beta = (\epsilon)^{4/\epsilon}, \alpha = \beta/C, q = 1/\alpha, \gamma = \epsilon^{4/\epsilon+2}$ in inequality (2), we have that

$$\frac{T_{1-\epsilon}(\beta)}{\beta} + \gamma/\beta \leq \epsilon + o(\epsilon)$$

and the following hardness statement of Small-Set-Expansion:

▶ **Theorem 10.** *For any $0 < \epsilon < 1$, and an arbitrarily large constant C, there exists some $k < n$ (functionally dependent on ϵ), for which it is* Small-Set-Expansion-hard *to distinguish the following two cases in a d-regular graph $G(V, E)$:*

- *there is a set $S \subseteq V$ of size k/C such that $\phi(S) \leq O(\epsilon)$*
- *every set $S \subseteq V$ of size less than k has expansion at least $1 - O(\epsilon)$*

3.2 Hardness Reduction

We shall make a gap preserving reduction from the Small-Set-Expansion hardness of Theorem 10. Given any d-regular graph with adjacency matrix A, will consider matrix M such that $M^{\mathrm{T}}M = I - \frac{1}{d}A$ for the RIP certification problem. Also without loss of generality, we can only prove for δ that is sufficiently small constant as if Theorem 4 holds for some $\delta = \delta_0$, it also holds for all $\delta \geq \delta_0$. In order to prove Theorem 4, it suffices to prove the following Lemma 11.

▶ **Lemma 11.** *Let $\delta = \epsilon^{0.4}$ for a sufficiently small constant ϵ. Then:*

1. *If there is a set S of size at most k/C and $\phi_G(S) \leq O(\epsilon)$, then the matrix is M not $(k/C, 1 - \delta)$-RIP*
2. *If for every set S of size at most k, $\phi_G(S) \geq 1 - O(\epsilon)$, then M is (k, δ)-RIP*

The proof of Lemma 11 would complete the proof of Theorem 4.

Proof. Given any d-regular graph with adjacency matrix A, let $x_S \in \{0,1\}^n$ be the indicator vector of a subset S. We know the number of edges that leave S is equal to $d \cdot |S| - x_S^{\mathrm{T}} A x_S = x_S^{\mathrm{T}}(d \cdot I - A)x_S$. Therefore, we have

$$\phi_G(S) = \frac{x_S^{\mathrm{T}}(d \cdot I - A)x_S}{d|S|} = \frac{x_S^{\mathrm{T}}(d \cdot I - A)x_S}{d\|x_S\|_2^2} = \frac{x_S^{\mathrm{T}}(I - A/d)x_S}{\|x_S\|_2^2} = \frac{x_S^{\mathrm{T}} M^{\mathrm{T}} M x_S}{\|x_S\|_2^2}$$

Let us prove the first claim. We know that when there is a set $S \subseteq V$ of size less than $\frac{k}{C}$ that has expansion less than $O(\epsilon)$. Let us denote $x_S \in \{0,1\}^n$ as the indicator of set S, then

$$\frac{x_S^{\mathrm{T}} \cdot M^{\mathrm{T}} M(x_S)}{\|x_S\|^2} \leq O(\epsilon)$$

which implies that

$$\|Mx\|_2 \leq O(\sqrt{\epsilon})\|x\|_2$$

Since x_S is k/C-sparse, after applying M, its length is only $O(\sqrt{\epsilon})$ times $\|x\|_2$. Now, given that we know $\delta = \epsilon^{0.4}$, for sufficiently small ϵ, we have that M is not $(k/C, \delta)$-RIP.

We shall prove the second claim of Lemma 11 by contradiction. Suppose there exists some k-sparse vector $x \in \mathbb{R}^n$ such that

$$\|Mx\|_2 \leq (1-\delta)\|x\|_2$$

We know then that

$$x^{\mathrm{T}} M^{\mathrm{T}} M x \leq (1-\delta)^2 \|x\|_2,$$

which implies that $\lambda_\delta(G) \leq 1 - 2\delta + \delta^2$. Now, by Theorem 7, we have that there must exist a set such that the expansion is at most

$$\sqrt{\lambda_\delta(G)(2 - \lambda_\delta(G))} \leq \sqrt{1 - (2\delta - \delta^2)^2} = 1 - \Theta(\epsilon^{0.8}),$$

which contradicts the fact that all sets S of size less than k must have expansion at least $1 - \epsilon$. ◀

4 Conclusion and Open Problems

In this paper, we establish that certifying RIP of a matrix (even approximately) is Small-Set-Expansion-hard in a strong sense. Although the Small-Set-Expansion problem is a conjecture, our work helps cement the place of RIP certification relative to other problems in regard to their hardness. In general, whenever we reduce from a known problem to a new problem, it increases the importance of the original problem.

One possible immediate open problems is to prove NP-hardness of RIP certification by reducing from known canonical problems. This would be interesting and important because the correctness of Small-Set-Expansion Hypothesis is uncertain. Another interesting direction to pursue could be to prove that RIP certification is hard even when the matrix satisfies certain natural properties such as coherence.

Acknowledgements. The authors would like to thank the anonymous reviewers for their constructive and insightful comments which helped in the final presentation of the paper.

References

1 Noga Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, 1986.
2 Afonso S. Bandeira, Edgar Dobriban, Dustin G. Mixon, and William F. Sawin. Certifying the restricted isometry property is hard. *IEEE Transactions on Information Theory*, 59(6):3448–3450, 2013.
3 Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.
4 Jean Bourgain, Stephen J Dilworth, Kevin Ford, Sergei V Konyagin, and Denka Kutzarova. Breaking the k^2 barrier for explicit rip matrices. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 637–644. ACM, 2011.
5 Emmanuel J Candes. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(9):589–592, 2008.
6 Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.
7 Emmanuel J Candes and Terence Tao. Decoding by linear programming. *Information Theory, IEEE Transactions on*, 51(12):4203–4215, 2005.
8 Mark A Davenport, Marco F Duarte, Yonina C Eldar, and Gitta Kutyniok. Introduction to compressed sensing. In *Compressed Sensing: Theory and Applications*. Cambridge University Press, 2012.
9 Ronald A DeVore. Deterministic constructions of compressed sensing matrices. *Journal of Complexity*, 23(4):918–925, 2007.
10 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? *SIAM Journal on Computing*, 37(1):319–357, 2007.
11 Pascal Koiran and Anastasios Zouzias. On the certification of the restricted isometry property. *CoRR*, abs/1103.4984, 2011.
12 Pascal Koiran and Anastasios Zouzias. Hidden cliques and the certification of the restricted isometry property. *CoRR*, abs/1211.0665, 2012.
13 George Pólya and Gabor Szego. *Isoperimetric inequalities in mathematical physics*. Princeton University Press, 1951.
14 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 755–764. ACM, 2010.
15 Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *Computational Complexity (CCC), 2012 IEEE 27th Annual Conference on*, pages 64–73. IEEE, 2012.
16 Alistair Sinclair and Mark Jerrum. Approximate counting, uniform generation and rapidly mixing markov chains. *Information and Computation*, 82(1):93–133, 1989.
17 David Steurer. Subexponential algorithms for d-to-1 two-prover games and for certifying almost perfect expansion. *Available at the author's website*, 1:2–1, 2010.
18 Terence Tao. Open question: deterministic uup matrices, July 2007.
19 Andreas M. Tillmann and Marc E. Pfetsch. The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing. *IEEE Transactions on Information Theory*, 60(2):1248–1259, 2014.
20 Luca Trevisan. Cs359g lecture 4: Spectral partitioning. CS359G Lecture 4: Spectral Partitioning.
21 Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Gap Amplification for Small-Set Expansion via Random Walks*

Prasad Raghavendra and Tselil Schramm

University of California, Berkeley
{prasad,tschramm}@cs.berkeley.edu

Abstract

In this work, we achieve gap amplification for the Small-Set Expansion problem. Specifically, we show that an instance of the Small-Set Expansion Problem with completeness ϵ and soundness $\frac{1}{2}$ is at least as difficult as Small-Set Expansion with completeness ϵ and soundness $f(\epsilon)$, for any function $f(\epsilon)$ which grows faster than $\sqrt{\epsilon}$. We achieve this amplification via random walks – the output graph corresponds to taking random walks on the original graph. An interesting feature of our reduction is that unlike gap amplification via parallel repetition, the size of the instances (number of vertices) produced by the reduction remains the same.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Gap amplification, Small-Set Expansion, random walks, graph products, Unique Games

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.381

1 Introduction

The small-set expansion problem refers to the problem of approximating the edge expansion of small sets in a graph. Formally, given a graph $G = (V, E)$ and a subset of vertices $S \subseteq V$ with $|S| \leq |V|/2$, the edge expansion of S is

$$\phi(S) = \frac{E(S, \bar{S})}{\mathrm{vol}(S)},$$

where $\mathrm{vol}(S)$ refers to the fraction of all edges of the graph that are incident on the subset S. The edge expansion of the graph G is given by $\phi_G = \min_{S \subseteq V, \mathrm{vol}(S) \leq 1/2} \phi(S)$. The problem of approximating the value of ϕ_G is the well-studied uniform sparsest cut problem [10, 4, 2].

In the small-set expansion problem, the goal is to approximate the edge expansion of the graph at a much finer granularity. Specifically, for $\delta > 0$ define the parameter $\phi_G(\delta)$ as follows:

$$\phi_G(\delta) = \min_{S \subseteq V, \mathrm{vol}(S) \leq \delta} \phi(S).$$

The problem of approximating $\phi_G(\delta)$ for all $\delta > 0$ is the small-set expansion problem.

The small-set expansion problem has received considerable attention in recent years due to its close connections to the unique games conjecture. To describe this connection, we will define a gap version of the problem.

* Prasad Raghavendra is supported by NSF Career Award and Alfred Sloan P. Fellowship. Tselil Schramm is supported by a Berkeley Chancellor's Fellowship and the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 1106400.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 381–391

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

▶ **Definition 1.** For constants $0 < s < c < 1$ and $\delta > 0$, the $SSE_\delta(c, s)$ problem is defined as follows: Given a graph $G = (V, E)$ distinguish between the following two cases:

- G has a set S with $\mathrm{vol}(S) \in [\delta/2, \delta]$ with expansion less than $1 - c$
- All sets S with $\mathrm{vol}(S) \leq \delta$ in G have expansion at least $1 - s$.

We will omit the subscript δ and write $SSE(c, s)$ when we refer to the $SSE_\delta(c, s)$ problem for all constant $\delta > 0$.

Recent work by Raghavendra and Steurer [13] introduced the following hardness assumption and showed that it implies the unique games conjecture.

▶ **Hypothesis 2.** *For all $\epsilon > 0$, there exists $\delta > 0$ such that $SSE_\delta(1 - \epsilon, \epsilon)$ is NP-hard.*

▶ **Theorem 3** ([13])**.** *The small set expansion hypothesis implies the unique games conjecture.*

Moreover, the small set expansion hypothesis is shown to be equivalent to a variant of the Unique Games Conjecture wherein the input instance is promised to be a small-set expander [14]. Assuming the small-set expansion hypothesis, hardness results have been obtained for several problems including Balanced Separator, Minimum Linear Arrangement [14] and the problem of approximating vertex expansion [11].

In this work, we will be concerned with gap amplification for the small set expansion problem. Gap amplification refers to an efficient reduction that takes a weak hardness result for a problem Π with a small gap between the completeness and soundness and produces a strong hardness with a much larger gap. Formally, this is achieved via an efficient reduction from instances of problem Π to *harder* instances of the same problem Π. Gap amplification is a crucial step in proving hardness of approximation results. An important example of gap amplification is the parallel repetition of 2-prover 1-round games or Label Cover. Label cover is a constraint satisfaction problem which is the starting point for a large number of reductions in hardness of approximation [7]. Starting with the PCP theorem, one obtains a weak hardness for label cover with a gap of 1 vs $1 - \beta_0$ for some tiny absolute constant β_0 [3]. Almost all label-cover based hardness results rely on the much stronger 1 vs ϵ hardness for label cover obtained by gap amplification via the parallel repetition theorem of Raz [16]. More recently, there have been significant improvements and simplifications to the parallel repetition theorem [15, 8, 5].

It is unclear if parallel repetition could be used for gap amplification for small set expansion. Given a graph G, the parallel repetition of G would consist of the product graph G^R for some large constant R. Unfortunately, the product graph G^R can have small non-expanding sets even if G has no small non-expanding sets. For instance, if G has a balanced cut then G^R could have a non-expanding set of volume $\frac{1}{2^R}$.

In this work, we show that random walks can be used to achieve gap amplification for small set expansion. Specifically, given a graph G the gap amplification procedure constructs G^t on the same set of vertices as G, but with edges corresponding to t-step lazy random walks in G. Using this approach, we are able to achieve the following gap amplification.

▶ **Theorem 4.** *Let f be any function such that $\lim_{\epsilon \to 0} \frac{f(\epsilon)}{\sqrt{\epsilon}} \to \infty$. Then:*

If for all $\epsilon > 0$, $SSE'(1 - \epsilon, 1 - f(\epsilon))$ is NP-hard then for all $\eta > 0$, $SSE(1 - \eta, 1/2)$ is NP-hard.

We remark here that the result has some discrepancy in the set sizes between the original instance and the instance produced by the reduction. For this reason, the reduction has to start with a slightly different version of the Small set expansion problem SSE' (See Definition 10).

The above result nicely complements the gap amplification result for the closely related problem of Unique Games obtained via parallel repetition [15]. For the sake of completeness we state the result below.

▶ **Theorem 5** ([15]). *Let f be any function such that $\lim_{\epsilon\to 0} \frac{f(\epsilon)}{\sqrt{\epsilon}} \to \infty$. Then:*
If for all $\epsilon > 0$ if $UG(1-\epsilon, 1-f(\epsilon))$ is NP-hard then for all $\eta > 0$, $UG(1-\eta, 1/2)$ is NP-hard.

Note that the size of the instance produced by our reduction remains bounded by $O(n^2)$. In fact, the instance produced has the same number of vertices but possibly many more edges. This is in contrast to parallel repetition wherein the size of the instance grows exponentially in the number of repetitions used.

Technically, the proof of the result is very similar to an argument in the work of Arora, Barak and Steurer [1] to show that graphs with sufficiently high threshold rank cannot be small-set expanders (see Steurer's thesis [17] for an improved version of the result). The work of O'Donnell and Wright [12] recast these arguments using continuous-time random walks instead of lazy-random walks, yielding cleaner and more general proofs. In this work, we will reuse the proof technique and obtain upper and lower bounds for the expansion profile of lazy random walks (see Theorem 11). These upper and lower bounds immediately imply the desired gap amplification result for small-set expansion.

Subsequent to our work, Kwok and Lau [9] have obtained a stronger analysis of our gap amplification theorem, yielding almost tight bounds.

2 Preliminaries

Unless otherwise specified, we will be concerned with an undirected graph $G = (V, E)$ with n vertices and associated edge weights $w : E \to \mathbb{R}^+$. The degree of vertex i denoted by $d(i) = \sum_{(i,j)\in E} w(i,j)$. The volume of a set $S \subseteq V$ is defined to be $\mathrm{vol}(S) = \sum_{i\in S} d(i)$. Henceforth, we will assume that the total volume is 1, i.e., $\sum_{i\in V} d(i) = 1$. The adjacency matrix A of the graph G has entries $A_{ij} = w(i,j)$. The degree matrix D is a $n \times n$ diagonal matrix with $D_{ii} = d(i)$.

2.1 Expansion Profile

The expansion profile of a graph is defined as follows.

▶ **Definition 6.** For a graph G, define the expansion profile $\phi_G : \mathbb{R}^+ \to [0,1]$ as

$$\phi_G(\delta) = \min_{S\subseteq V, \mathrm{vol}(S)\leq\delta} \phi(S)$$

where $\phi(S) = \frac{E(S,\bar{S})}{\mathrm{vol}(S)}$.

2.2 Lazy Random Walks

The transition matrix for a lazy random walk on G is given by

$$M = \tfrac{1}{2}(I + D^{-1}A)\,.$$

The lazy random walk corresponds to staying at the same vertex with probability $\frac{1}{2}$, and moving to a random neighbor with probability $\frac{1}{2}$. We will let G^t denote the graph corresponding to the t-step lazy random walk. The adjacency matrix of G^t is given by DM^t.

We recall a few standard facts about lazy random walks here.

▶ **Fact 7.** *If G is a graph with adjacency matrix A, then G's lazy random walk operator $M = \frac{1}{2}(I + D^{-1}A)$ has the property that $\|D^{1/2}Mv\|_2^2 = v^T DM^2 v$ for any vector v.*

Proof. We use the fact that $M = \frac{1}{2}D^{-1/2}(I + D^{-1/2}AD^{-1/2}))D^{1/2}$:

$$\begin{aligned}\|D^{1/2}Mv\|_2^2 &= \tfrac{1}{4}v^T M^T DMv \\ &= \tfrac{1}{4}v^T D^{1/2}(I + D^{-1/2}AD^{-1/2})D^{-1/2}DD^{-1/2}(I + D^{-1/2}AD^{-1/2})D^{1/2}v \\ &= \tfrac{1}{4}v^T D^{1/2}(I + D^{-1/2}AD^{-1/2})^2 D^{1/2}v \\ &= v^T DM^2 v,\end{aligned}$$

as desired. ◀

▶ **Fact 8.** *If G is a graph with adjacency matrix A, then for the lazy random walk operator $M = \frac{1}{2}(I + D^{-1}A)$, we have*

$$\|D^{1/2}v\|_2^2 = v^T Dv \geq v^T DMv \geq v^T DM^2 v = \|D^{1/2}Mv\|_2^2.$$

Proof. Since the eigenvalues λ_i of $D^{-1/2}AD^{-1/2}$ are between $[-1, 1]$, the eigenvalues of $M' = \frac{1}{2}(I + D^{-1/2}AD^{-1/2})$ are $\mu_i = \frac{1}{2}(1 + \lambda_i)$, and so $\mu_i \in [0, 1]$. Let $D^{1/2}v = \sum \alpha_i u_i$ be the decomposition of $D^{1/2}v$ in terms of the eigenvectors of M'. Then we have

$$D^{1/2}Mv = M'D^{1/2}v = \sum \alpha_i \mu_i u_i,$$

and so $v^T Dv = \sum \alpha_i^2$, $v^T DMv = \sum \alpha_i^2 \mu_i$, and $v^T DM^2 v = \sum \alpha_i^2 \mu_i^2$. Since $\mu_i \in [0, 1]$, we have $v^T Dv \geq v^T DMv \geq v^T DM^2 v$, as desired. ◀

▶ **Fact 9.** *For the lazy random walk operator $M = \frac{1}{2}(I + D^{-1}A)$ and any vector $v \in \mathbb{R}^V$, $v \geq 0$ we have*

$$\|Dv\|_1 = \|DMv\|_1.$$

Proof. Let $v \in \mathbb{R}^V$. We have

$$\|DMv\|_1 = 1^T D(\tfrac{I+D^{-1}A}{2})v \;=\; \tfrac{1}{2}((1^T D)v + (1^T A)v) \;=\; \|Dv\|_1,$$

where the last inequality follows because $1^T D = 1^T A$. ◀

2.3 Small-Set Expansion Problem

The formal statement of the SSE' problem is as follows.

▶ **Definition 10.** For constants $0 < s < c < 1$ and $\delta > 0$, the Small-Set Expansion problem $SSE'_\delta(c, s)$ is defined as follows: Given a graph $G = (V, E)$, distinguish between the following two cases:

- G contains a set S such that $\mathrm{vol}(S) \in [\delta/2, \delta]$ and $\phi(S) \leq 1 - c$
- All sets S with $\mathrm{vol}(S) \leq \mathbf{8}\delta$ in G have expansion $\phi(S) \geq 1 - s$.

The key difference from $SSE_\delta(c, s)$ is that the soundness is slightly stronger in that even sets of size 8δ have expansion at least $1 - s$.

2.4 Organization

In Section 3, we will obtain upper and lower bounds (Theorem 11) for expansion profile of lazy random walks. Subsequently, we use these bounds to conclude the main result of the paper in Section 4. In Appendix A, we give a reduction that establishes the equivalence of the search versions of two different notions of Small-Set Expansion. Finally, we also present a reduction from SSE on irregular graphs to SSE on regular graphs in Appendix B.

3 Expansion Profile of Lazy Random Walks

Let $G = (V, E)$ be a graph with adjacency matrix A, and diagonal degree matrix D. The transition matrix for a lazy random walk on G is $M = \frac{1}{2}(I + D^{-1}A) = \frac{1}{2}D^{-1/2}(I + D^{-1/2}AD^{-1/2})D^{1/2}$.

For every $t \in \mathbb{N}$, let G^t denote the graph corresponding to the t-step lazy random walk whose adjacency matrix is given by DM^t. We will prove the following theorem about the expansion profile of G^t.

▶ **Theorem 11.** *For all $t \in \mathbb{N}$ and $\eta, \delta \in (0, 1]$, if G^t denotes the graph corresponding to the t-step lazy random walk in a graph $G = (V, E)$ then,*

$$\min\left(1 - \left(1 - \frac{\phi_G^2(\frac{4\delta}{\eta})}{32}\right)^t, 1 - \eta\right) \leq \phi_{G^t}(\delta) \leq \frac{t}{2} \cdot \phi_G(\delta)\,.$$

We will split the proof of the above theorem in to two parts: Lemma 12 and Lemma 13

▶ **Lemma 12.** *For every subset $S \subseteq V$,*

$$\phi_{G^t}(S) \leq \frac{t}{2} \cdot \phi_G(S),$$

and therefore $\phi_{G^t}(\delta) \leq \frac{t}{2} \cdot \phi_G(\delta)$.

Proof. Fix a subset $S \subset V$. From [6], we have that the probability $p(t)$ that a lazy random walk stays entirely in S for t steps is bounded below by

$$p(t) \geq \left(1 - \frac{1}{2}\phi(S)\right)^t.$$

Now, the expansion of S in G^t is the probability of leaving the set on the tth step of the random walk, which is at most $1 - p(t)$. Hence,

$$\phi_{G^t}(S) \leq 1 - p(t) \leq 1 - \left(1 - \frac{1}{2}\phi(S)\right)^t \leq \frac{t}{2}\phi(S),$$

as desired. The result immediately follows for all sets of volume $\leq \delta$. ◀

▶ **Lemma 13.** *For all t, η,*

$$\phi_{G^t}(\delta) \geq \min\left(1 - \left(1 - \frac{\phi_G^2(\frac{4\delta}{\eta})}{32}\right)^t, 1 - \eta\right).$$

We prove this lemma by contradiction, by showing that if the expansion in the final graph is not large enough then there exists a vector with bounded Rayleigh quotient with respect to the original graph, from which we can extract a non-expanding set. The intuition is that the expansion of a set in the final graph DM^t corresponds to the neighborhood of the random walk after t steps, and if the neighborhood is not large enough after t steps, there must be at least one step (or application of M) during which it did not grow.

Proof. Suppose by way of contradiction that this is not the case. Let $\beta = \phi_G(\frac{4\delta}{\eta})$ and let $\delta' = \frac{4\delta}{\eta}$. Further, let $\hat{\beta} = \frac{1}{2}\beta$.

Let S be a set of volume at most $\delta \cdot \mathrm{vol}(V)$ such that

$$\phi_{G^t}(S) \leq \min\left(1 - \left(1 - \frac{\hat{\beta}^2}{8}\right)^t, 1 - \eta\right). \tag{1}$$

Let $v_0 = 1_S$ be the vector corresponding to the indicator function of the set S. Define $v_i = M^i v_0$, and for the diagonal degree matrix D of A, define $w_i = D^{1/2} v_i$. Note that $\|w_0\|_2^2 = \mathrm{vol}(S)$, and $\|Dv_0\|_1 = \mathrm{vol}(S)$. By Fact 9 we also have $\|Dv_i\|_1 = \mathrm{vol}(S)$ for all i.

We first lower-bound $\|w_{\frac{t}{2}}\|_2$. By definition of expansion,

$$\phi_{G^t}(S) = 1 - \frac{v_0^T D M^t v_0}{v_0^T D v_0}$$

which by Fact 7 implies that $\|D^{1/2} M^{\frac{t}{2}} v_0\|_2^2 = \mathrm{vol}(S)(1 - \phi_{G^t}(S))$. Now, using (1) we get

$$\|w_{\frac{t}{2}}\|_2^2 \;=\; \|D^{1/2} M^{\frac{t}{2}} v_0\|_2^2 = \mathrm{vol}(S)(1 - \phi_{G^t}(S)) \;\geq\; \mathrm{vol}(S) \cdot \max\left(\eta, (1 - \frac{1}{8}\hat{\beta}^2)^t\right) \tag{2}$$

By Fact 8, we have $\|w_i\|_2 \geq \|w_{i+1}\|_2 \geq 0$ for all i, and (2) holds for all $i \leq \frac{t}{2}$.

We now assert that there must be some i for which

$$\frac{\|w_{i+1}\|_2^2}{\|w_i\|_2^2} > 1 - \tfrac{1}{4}\hat{\beta}^2.$$

To see this, consider the product of all such terms for $i < \frac{t}{2}$. Some algebraic simplification shows that

$$\prod_{i=0}^{\frac{t}{2}-1} \frac{\|w_{i+1}\|_2^2}{\|w_i\|_2^2} \;=\; \frac{\|w_{\frac{t}{2}}\|_2^2}{\|w_0\|_2^2} \;>\; \frac{(1 - \frac{1}{8}\hat{\beta}^2)^t \cdot \mathrm{vol}(S)}{\mathrm{vol}(S)} \;=\; \left(1 - \tfrac{1}{8}\hat{\beta}^2\right)^t,$$

where the second-to-last inequality follows from (2). Thus for some $i < \frac{t}{2}$ we have

$$\frac{\|w_{i+1}\|_2^2}{\|w_i\|_2^2} \;>\; \left((1 - \frac{1}{8}\hat{\beta}^2)^t\right)^{\frac{2}{t}} \;>\; 1 - \frac{1}{4}\hat{\beta}^2.$$

Then let w_i be the vector corresponding to the first i for which $\|w_{i+1}\|_2^2 \geq (1 - \frac{1}{4}\hat{\beta}^2)\|w_i\|_2^2$.

Since w_{i+1} is obtained from v_i via one step of a lazy random walk and a normalization, we can bound the Rayleigh quotient of v_i with respect to the Laplacian of $DM = \frac{1}{2}(D + A)$:

$$\frac{v_i^T D(I - M)v_i}{v_i^T D v_i} = 1 - \frac{v_i^T DM v_i}{v_i^T D v_i},$$

by Fact 8,

$$\leq 1 - \frac{v_i^T DM^2 v_i}{v_i^T D v_i}$$

and by Fact 7,

$$\begin{aligned} &= 1 - \frac{\|w_{i+1}\|_2^2}{\|w_i\|_2^2} \\ &\leq \frac{1}{4}\hat{\beta}^2. \end{aligned} \tag{3}$$

We now truncate the vector v_i, then run Cheeger's algorithm on the truncated vector in order to find a non-expanding small set, and thus obtain a contradiction. Let $\theta = \frac{\eta}{4}$. We take the truncated vector

$$z_i(j) = \begin{cases} v_i(j) - \theta & v_i(j) \geq \theta \\ 0 & \text{otherwise} \end{cases} .$$

By Fact 9, Dv_i has L_1 mass $\mathrm{vol}(S)$. Thus, the total volume of the set S_z of vertices with nonzero support in z_i is

$$\mathrm{vol}(S_z) \;=\; \sum_{v_i(j)>\theta} d(j) \;\leq\; \sum_{v_i(j)>\theta} \frac{1}{\theta} d(j) v_i(j) \;\leq\; \frac{1}{\theta} \cdot \|Dv_i\|_1 = \frac{4\,\mathrm{vol}(S)}{\eta}$$

Hence any subset of S_z has volume at most $\frac{4\,\mathrm{vol}(S)}{\eta}$.

For the vector v_i, we know that $\|Dv_i\|_1 = \mathrm{vol}(S)$. Moreover using (2),

$$\|D^{1/2} v_i\|_2^2 = \|w_i\|_2^2 \geq \|w_{t/2}\|_2^2 \geq \eta\,\mathrm{vol}(S)\,.$$

Applying Lemma 14 to v_i and z_i to conclude,

$$\frac{z_i^T D(I-M) z_i}{z_i^T D z_i} \leq 2 \frac{v_i^T D(I-M) v_i}{v_i^T D v_i}\,.$$

Using (3), this implies the following bound on the Rayleigh quotient of z_i,

$$\frac{z_i^T D(I-M) z_i}{z_i^T D z_i} \;\leq\; \frac{1}{2}\hat{\beta}^2\,.$$

Thus, when we run Cheeger's algorithm on z_i, we get a set of volume at most $\frac{4\,\mathrm{vol}(S)}{\eta}$ and of expansion less than $\hat{\beta}$ in DM, and therefore less than β in G. Since $\beta = \phi_G(\frac{4\delta}{\eta})$, this is a contradiction. This completes the proof of Lemma 13. ◀

The following lemma, which gives an upper bound on the Rayleigh quotient of a truncated vector, is a slight generalization of Lemma 3.4 of [1].

▶ **Lemma 14.** *Let $x \in \mathbb{R}^V$ be non-negative, let L be the weighted Laplacian of a graph $G = (V, E)$ with weights $w(i,j)$ and degree matrix D. Suppose that*

$$4\theta \|Dx\|_1 \leq \|D^{1/2} x\|_2^2 \tag{4}$$

Then for the threshold vector y defined by

$$y(i) = \begin{cases} x(i) - \theta & x(i) > \theta \\ 0 & \textit{otherwise} \end{cases} ,$$

we have

$$\frac{y^T L y}{y^T D y} \;\leq\; 2 \cdot \frac{x^T L x}{x^T D x}.$$

Proof. First, we show $y^T L y \leq x^T L x$.

$$\begin{aligned} y^T L y \;&=\; \sum_{(i,j)\in E} w(i,j)(y(i)-y(j))^2 \\ &=\; \sum_{\substack{(i,j)\in E \\ y(i),y(j)\geq 0}} w(i,j)(x(i)-x(j))^2 + \sum_{\substack{(i,j)\in E \\ y(i)\geq 0, y(j)=0}} w(i,j)(x(i)-\theta)^2 \\ &\leq\; \sum_{(i,j)\in E} w(i,j)(x(i)-x(j))^2 \\ &=\; x^T L x, \end{aligned}$$

where the second-to-last inequality follows from the fact that if $y(i) = 0$, then $x(i) \leq \theta$.

Now, we show that $y^T Dy \geq \frac{1}{2} x^T Dx$. First, we note that $d(i)y(i)^2 \geq d(i)x(i)^2 - 2\theta d(i)x(i)$ for all k. Thus,

$$\begin{aligned}\sum_{i\in V} d(i)y(i)^2 &\geq \sum_{i\in V} d(i)x(i)^2 - 2\theta d(i)x(i) \\ &= \left(\sum_{i\in V} d(i)x(i)^2\right) - 2\theta\left(\sum_{i\in V} d(i)x(i)\right) \\ &\geq \frac{1}{2}\sum_{i\in V} d(i)x(i)^2 .\end{aligned}$$

Where the the last inequality follows by assumption (4).

Thus, we have

$$\frac{y^T Ly}{y^T Dy} \leq 2 \cdot \frac{x^T Lx}{x^T Dx},$$

as desired. ◀

4 Gap Amplification

In this section, we will prove Theorem 4 which we restate here for convenience.

▶ **Theorem 15** (Restatement of Theorem 4). *Let f be any function such that $\lim_{\epsilon\to 0} \frac{f(\epsilon)}{\sqrt{\epsilon}} \to \infty$. Then:*

If for all $\epsilon > 0$, $SSE'(1-\epsilon, 1-f(\epsilon))$ is NP-hard then for all $\eta > 0$ $SSE(1-\eta, \frac{1}{2})$ is NP-hard.

Proof. Fix ϵ small enough so that $\frac{64\epsilon}{f(\epsilon)^2} \leq \eta$. There exists such an ϵ since $\lim_{\epsilon\to 0} \frac{f(\epsilon)}{\sqrt{\epsilon}} \to \infty$. Fix $t = \frac{64}{f(\epsilon)^2}$.

Given an instance G of $SSE'(1-\epsilon, 1-f(\epsilon))$, the reduction just outputs the graph G^t obtained via t-step lazy random walks on G. Since the adjacency matrix of G' can be calculated with $\log t$ matrix multiplications, this reduction clearly runs in time $O(n^3 \log t)$.

Completeness. If there exists a set of S with $\mathrm{vol}(S) \in [\delta/2, \delta]$ and $\phi_G(S) \leq \epsilon$ then by Lemma 12 the same set S satisfies,

$$\phi_{G^t}(S) \leq \frac{t}{2}\phi_G(S) = \Theta\left(\frac{\epsilon}{f(\epsilon)^2}\right) \leq \eta .$$

Soundness. If $\phi_G(8\delta) \geq f(\epsilon)$ then by applying Lemma 13

$$\phi_{G^t}(\delta) \geq \min\left(1 - \left(1 - \frac{1}{32} f(\epsilon)^2\right)^t, 1/2\right) \geq \frac{1}{2} .$$

◀

References

1 Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. In *FOCS*, pages 563–572, 2010.

2 Sanjeev Arora, James R. Lee, and Assaf Naor. Euclidean distortion and the sparsest cut. *Journal of American Mathematical Society*, 21:1–21, 2008.

3 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *JACM: Journal of the ACM*, 45, 1998.

4 Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows, geometric embeddings and graph partitioning. In *Proceedings of the thirty-sixth annual ACM Symposium on Theory of Computing (STOC-04)*, pages 222–231, New York, June 13–15 2004. ACM Press.

5 Irit Dinur and David Steurer. Analytical approach to parallel repetition. *CoRR*, abs/1305.1979, 2013.

6 Shayan Oveis Gharan and Luca Trevisan. Approximating the expansion profile and almost optimal local graph clustering. In *FOCS*, pages 187–196, 2012.

7 Johann Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.

8 Holenstein. Parallel repetition: Simplifications and the no-signaling case. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 2007.

9 Tsz Chiu Kwok and Lap Chi Lau. Personal communication, 2014.

10 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.

11 Anand Louis, Prasad Raghavendra, and Santosh Vempala. The complexity of approximating vertex expansion. *CoRR*, abs/1304.3139, 2013.

12 Ryan O'Donnell and David Witmer. Markov chain methods for small-set expansion. *Arxiv arXiv:1204.4688*, 2012.

13 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *STOC*, pages 755–764, 2010.

14 Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *IEEE Conference on Computational Complexity*, pages 64–73, 2012.

15 Anup Rao. Parallel repetition in projection games and a concentration bound. In Richard E. Ladner and Cynthia Dwork, editors, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC'08)*, pages 1–10. ACM, 2008.

16 Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, June 1998.

17 David Steurer. *On the Complexity of Unique Games and Graph Expansion.* PhD thesis, Princeton University, 2010.

A Equivalence of Two Notions of the Small-Set Expansion Problem

There is a slightly different version of the Small-Set expansion decision problem that differs from Definition 1 in the soundness case.

▶ **Definition 16.** For constants $0 < s < c < 1$, and $\delta > 0$, the Small-Set Expansion problem $SSE^{=}_{\delta}(c, s)$ is defined as follows: Given a graph $G = (V, E)$ with $\mathrm{vol}(V) = N$, distinguish between the following two cases:

- G has a set of volume in the range $[\frac{1}{2}\delta N, \delta N]$ with expansion less than $1 - c$
- All sets in G of volume in the range $[\frac{1}{4}\delta N, \delta N]$ have expansion at least $1 - s$.

Clearly $SSE^{=}_{\delta}(c,s)$ is a harder decision problem than $SSE_{\delta}(c,s)$ since the soundness assumption is weaker. There is no known reduction from $SSE^{=}_{\delta}(c,s)$ to $SSE_{\delta}(c,s)$ that establishes the equivalence of the two versions. Here we observe that the search versions of these two problems are equivalent.

▶ **Proposition 17.** *For all $\delta_0, c, s > 0$ A search algorithm for $SSE_{\delta}(2c-1,s)$ for $\delta \in [\delta_0/2, \delta_0]$ gives a search algorithm for $SSE^{=}_{\delta}(c,s)$ in the range $\delta \in [\delta_0/2, \delta_0]$.*

Proof. Suppose we are given an algorithm A that finds a set S' of volume at most δN and expansion less than $1-s$ whenever there exists a set S with $\mathrm{vol}(S) \in [\frac{1}{4}\delta N, \delta N]$ and $\Phi(S) \leq 2-2c$. We construct a set $S \subseteq V$ such that $\mathrm{vol}(S) \in [\frac{1}{4}\delta N, \delta N]$ and $\phi(S) < 1-s$. We proceed iteratively, as follows.

We start with an empty initial set, S_{out}, and with the full graph, $G_0 = G$. If $\mathrm{vol}(S_{out}) \in [\frac{1}{4}\delta N, \delta N]$, we terminate and return S_{out}. Otherwise, at the ith step, we apply A to G_{i-1} to obtain a set S_i of expansion less than $1-s$. If $\mathrm{vol}(S_i) \in [\frac{1}{4}\delta N, \delta N]$ return S_i, otherwise add the vertices in S_i to S_{out}. We then set $G_i = G_{i-1} \setminus S_i$. If no such set can be found, then we terminate and return no.

Clearly, this algorithm terminates and runs in polynomial time. Suppose S' is a non-expanding set with $\mathrm{vol}(S') \in [\frac{1}{2}\delta N, \delta N]$. As long as S_{out} has volume smaller than $\frac{1}{4}\delta N$, $S' - S_{out}$ will have volume at least $\mathrm{vol}(S')/2$ and has expansion at most $2\phi(S') \leq 2-2c$. Hence by the assumption about algorithm A, it will return a set S_i of expansion at most $1-s$. The check of the volume of S_i ensures that S_{out} will never go from below the allowable volume range to above in a single step. Finally if S_i was never returned for any step i, the union of all the sets S_i has expansion at most $1-s$ and volume in the range $[\delta N/4, \delta N]$. ◀

B Reduction from Irregular Graphs to Regular Graphs

In this section, we present a reduction from small set expansion on irregular graphs to small set expansion on regular graphs. Specifically, we prove the following theorem.

▶ **Theorem 18.** *There exists an absolute constant C such that for all $\gamma, \beta \in (0,1)$ there is a polynomial time reduction from $SSE_{\delta}(1-\gamma, 1-\beta)$ on a irregular graph $G = (V,E)$ to $SSE_{\delta}(1-\gamma, 1-\beta/C)$ on a 4-regular graph $G' = (V', E')$*

Proof. The reduction is as follows: we replace each vertex $v \in V$ with a 3-regular expander A_v on $\deg(v)$ vertices. Using standard constructions of 3-regular expanders, we can assume that the graphs A_v have edge expansion at least $\kappa = 0.01$. Now, for each edge $(v,w) \in E$, we add an edge between a particular vertex in A_v and A_w. The resulting graph on the expanders is G', with $V' = \cup_{v \in V} A_v$. Note that G' is d-regular, and that $|V'| = \sum_{v \in V} \deg(v) = \mathrm{vol}(V)$, as desired.

For the completeness, we note that if a set $S \subset V$ with volume at most $\delta|V|$ has $\phi_G(S) < \gamma$, then the set $S' = \cup_{v \in S} A_v$ has the same number of edges leaving the set as S, and the number of vertices in the set is equal to $\mathrm{vol}(S)$. Thus, $\phi_{G'}(S') < \gamma/4$, as desired.

For soundness, suppose there is a set $S' \subset V'$ with $|S'| \leq \delta|V'|$ and $\phi_{G'}(S') < \beta$. Then we can partition S' into sets corresponding to each A_v; let $B_v = S' \cap A_v$. Then consider the set

$$S^* = \cup_{|B_v| \geq \frac{1}{2}|A_v|} A_v,$$

the set of A_v that overlap with S' by at least half. We will argue that S^* has expansion at

most $\frac{10}{\kappa}\beta$ in G'. First, by definition of expansion we have

$$\beta \geq \phi_{G'}(S') = \frac{\sum_v E(B_v, \bar{S'})}{4\sum_{v\in V}|B_v|} = \frac{\sum_v E(B_v, A_v \setminus B_v) + E(B_v, \bar{S'} \setminus A_v)}{4\sum_{v\in V}|B_v|},$$

where we distinguish between boundary edges of S' inside and outside of the A_v. In particular, we have

$$4\beta \sum_{v\in V}|B_v| \geq \sum_{v\in V} E(B_v, A_v \setminus B_v).$$

Now, we bound from below the number of boundary edges within A_v. Since A_v is an expander with expansion κ, we have

$$E(B_v,\ A_v \setminus B_v) \geq \kappa \cdot \min(|B_v|, |A_v \setminus B_v|).$$

Hence we will have,

$$S'\Delta S^* = \sum_v \min(|B_v|, |A_v \setminus B_v|) \leq \frac{1}{\kappa}\sum_v E(B_v, A_v \setminus B_v) \leq \frac{4\beta}{\kappa}\sum_{v\in V}|B_v| = \frac{4\beta}{\kappa}|S'|\,.$$

Since G' is a 4-regular graph, we can upper bound the expansion of S^* by

$$\phi_{G'}(S^*) \leq \frac{E[S', \bar{S'}] + 4|S'\Delta S^*|}{4|S'| - 4|S'\Delta S^*|} \leq \frac{4\beta|S'| + 16\beta/\kappa|S'|}{4|S'| - 16\beta/\kappa|S'|} \leq \frac{\beta\,(1 + 4/\kappa)}{1 - 4\beta/\kappa}\,.$$

Thus, in G the set $S = \{v \mid A_v \in S^*\}$ has expansion at most $\frac{10}{\kappa}\beta$, and $\mathrm{vol}(S) \in [\frac{1}{2}\delta\,\mathrm{vol}(V), 2\delta\,\mathrm{vol}(V)]$, as desired. ◀

Power of Preemption on Uniform Parallel Machines

Alan J. Soper and Vitaly A. Strusevich

Department of Mathematical Sciences, University of Greenwich
Old Royal Naval College, Park Row, Greenwich, London, SE10 9LS, U.K.
{A.J.Soper,V.Strusevich}@greenwich.ac.uk

Abstract

For a scheduling problem on parallel machines, the power of preemption is defined as the ratio of the makespan of an optimal non-preemptive schedule over the makespan of an optimal preemptive schedule. For m uniform parallel machines, we give the necessary and sufficient conditions under which the global bound of $2 - 1/m$ is tight. If the makespan of the optimal preemptive schedule is defined by the ratio of the total processing times of $r < m$ longest jobs over the total speed of r fastest machines, we show that the tight bound on the power of preemption is $2 - 1/\min\{r, m - r\}$.

1998 ACM Subject Classification F.2.2 Sequencing and Scheduling, G.2.1 Combinatorial Algorithms, G.1.6.Optimization

Keywords and phrases Machine Scheduling, Uniform Parallel Machines, Power of Preemption

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.392

1 Introduction

In this paper, we perform an analysis of the power of preemption for scheduling problems on uniform parallel machines.

In parallel machine scheduling, we are given the jobs of the set $N = \{J_1, J_2, \ldots, J_n\}$ and m parallel machines M_1, $M_2, \ldots, M_m$. If a job $J_j \in N$ is processed on machine M_i alone, then its processing time is known to be p_{ij}. There are three main types of scheduling systems with parallel machines: (i) *identical* parallel machines, for which the processing times are machine-independent, i. e., $p_{ij} = p_j$; (ii) *uniform* parallel machines, which have different speeds, so that $p_{ij} = p_j/s_i$, where s_i denotes the *speed* of machine M_i; and (iii) *unrelated* parallel machines, for which the processing time of a job depends on the machine assignment.

In all problems considered in this paper the objective is to minimize the *makespan,* i. e., the maximum completion time. For a schedule S, the makespan is denoted by $C_{\max}(S)$. In a non-preemptive schedule, each job is processed on the machine it is assigned to without interruption. In a preemptive schedule, the processing of a job on a machine can be interrupted at any time and then resumed either on this or on any other machine, provided that the job is not processed on two or more machines at a time. For an instance of a scheduling problem on parallel machines, let S^*_{np} and S^*_p denote an optimal non-preemptive and an optimal preemptive schedule, respectively.

The problem of finding an optimal non-preemptive schedule on identical parallel machines is NP-hard, and the corresponding problems on uniform or unrelated machines are obviously no easier. The preemptive counterparts of these problems are polynomially solvable, even in the most general settings with unrelated machines. See a focused survey [3] on parallel machine scheduling with the makespan objective for details and references.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 392–402

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

Consider an instance of a scheduling problem to minimize the makespan $C_{\max}$ on m parallel machines (identical, uniform or unrelated). For the corresponding problem, we define the *power of preemption* as the maximum ratio $C_{\max}(S^*_{np})/C_{\max}(S^*_p)$ across all instances of the problem at hand. We denote the power of preemption by ρ_m. The power of preemption determines what can be gained regarding the maximum completion time if preemption is allowed.

In order to determine the exact value of ρ_m for a particular problem and to give that concept some practical meaning, the following should be done:

(i) demonstrate that the inequality

$$\frac{C_{\max}\left(S^*_{np}\right)}{C_{\max}\left(S^*_p\right)} \leq \rho_m \tag{1}$$

holds for all instances of the problem;

(ii) exhibit instances of the problem for which (1) holds as equality, i. e., to show that the value of ρ_m is tight; and

(iii) develop a polynomial-time algorithm that finds a heuristic non-preemptive schedule S_{np} such that

$$\frac{C_{\max}\left(S^*_{np}\right)}{C_{\max}\left(S^*_p\right)} \leq \frac{C_{\max}\left(S_{np}\right)}{C_{\max}\left(S^*_p\right)} \leq \rho_m. \tag{2}$$

If the machines are identical parallel, then it is known that $\rho_m = 2 - 2/(m+1)$, as independently proved in [1] and [9]. It is shown in [11], that the value of ρ_m can be reduced for some instances that contain jobs with fairly large processing times.

For unrelated parallel machines, a rounding procedure that is attributed to Shmoys and Tardos and reproduced in [10] and [4] finds non-preemptive schedules S_{np} such that the bound (2) holds for $\rho_m = 4$. This bound is tight, as proved in [4].

According to [13], for uniform parallel machines $\rho_m = 2 - 1/m$. For $m = 2$ a parametric analysis of the power of preemption with respect to the speed of the faster machine is independently performed in [7] and [12]. For $m = 3$, a similar analysis is contained in [12], provided that the machine speeds take at most two values, 1 and $s \geq 1$.

2 Preliminaries

An instance I of the problem with n jobs and m parallel uniform machines is defined by the list $\mathcal{L}_n = (p_1, p_2, \ldots, p_n)$ of the processing times of the jobs and the list $\mathcal{M}_m = (s_1, s_2, \ldots, s_m)$ of the machine speeds. The machines are numbered in non-increasing order of their speeds, i. e., $s_1 \geq s_2 \geq \cdots \geq s_m$.The jobs are numbered in accordance with the following truncated LPT rule, i. e., m longest jobs are numbered in non-increasing order of their processing times

$$p_1 \geq p_2 \geq \cdots \geq p_m, \tag{3}$$

while the remaining jobs, all at lest as short as p_m, are numbered arbitrary.

Feasible non-preemptive and preemptive schedules for an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ are denoted by $S_{np}(\mathcal{L}_n, \mathcal{M}_m)$ or $S_{np}(I)$, and by $S_p(\mathcal{L}_n, \mathcal{M}_m)$ or $S_p(I)$, respectively; the corresponding optimal non-preemptive and preemptive schedules are denoted by $S^*_{np}(\mathcal{L}_n, \mathcal{M}_m)$ or $S^*_{np}(I)$ and by $S^*_p(\mathcal{L}_n, \mathcal{M}_m)$ or $S^*_p(I)$, respectively. The reference to an instance may be omitted if it is clear which instance is being discussed.

In our analysis of the power of preemption, we will need precise expressions for the makespan of the preemptive schedules. The fastest algorithm for finding an optimal preemptive schedule on uniform parallel machines is due to Gonzalez and Sahni [6] and requires $O(n + m \log m)$ time.

Given an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$, for each u, $1 \leq u \leq m$, define the total speed of the u fastest machines $S_u = \sum_{i=1}^{u} s_i$. Besides, define the set of u longest jobs $H_u = \{1, 2, \ldots, u\}$, and for a set of jobs $Q \subseteq N$, define $p(Q) = \sum_{j \in Q} p_j$, where for completeness $p(\emptyset) = 0$.

It is well-known (see, e.g., [2]) that for an optimal preemptive schedule $S_p^*(I)$ the makespan is equal to

$$C_{\max}(S_p^*(I)) = \max\{T_u | 1 \leq u \leq m\}, \tag{4}$$

where

$$T_u = p(H_u)/S_u,\ 1 \leq u \leq m-1;\ T_m = p(N)/S_m. \tag{5}$$

In our consideration, we classify the instances on m uniform machines as follows.

▶ **Definition 1.** An instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ is said to belong to Class r, $1 \leq r \leq m$, if $C_{\max}\left(S_p^*(I)\right) = T_r = \max\{T_u | 1 \leq u \leq m\}$.

Notice that an instance may belong to several classes simultaneously, if there is a tie for the maximum value of T_u, $1 \leq u \leq m$.

A non-preemptive schedule $S_{np}(I)$ is defined by a partition of set N into m subsets N_1, $N_2, \ldots, N_m$, where the jobs of set N_i and only those are assigned to be processed on machine M_i, $1 \leq i \leq m$. Notice that even in an optimal schedule some of these subsets can be empty.

A popular heuristic for finding a non-preemptive schedule on uniform parallel machines is known as the LPT List Scheduling. According to this algorithm, the jobs are scanned in accordance with the LPT rule, i. e., in non-increasing order of their processing times, the next job is assigned to the machine where it will complete as early as possible. For an instance I on uniform machines, let the LPT algorithm output a schedule $S(I)$. It can be found in $O(nm + n \log n)$ time. The best known results on the worst-case ratio $\rho_{LPT} = C_{\max}(S(I))/C_{\max}\left(S_{np}^*(I)\right)$ are due to Kovacs [8] who proves $1.54 \leq \rho_{LPT} \leq 1.577$. It is proved in [13] that $C_{\max}(S(I))/C_{\max}\left(S_p^*(I)\right) \leq 2 - 1/m$, and this bound is tight. For a preemptive schedule $S_p(I)$ found by a preemptive modification of the LPT algorithm the inequality $C_{\max}(S_p(I))/C_{\max}\left(S_p^*(I)\right) \leq 2 - 2/(m+1)$ holds; see [5].

In the subsequent sections, we only consider instances in which the number of jobs is no smaller than the number of machines. Take an instance $(\mathcal{L}_n, \mathcal{M}_m)$ with $n < m$. Let $\mathcal{M}_n$ be the list of machine speeds obtained from list $\mathcal{M}_m$ by a removal of the $m - n$ slowest machines.

It is clear that in each schedule $S_{np}^*(\mathcal{L}_n, \mathcal{M}_m)$ and $S_p^*(\mathcal{L}_n, \mathcal{M}_m)$ the jobs are assigned to at most n fastest machines. Thus, in the non-preemptive case, $S_{np}^*(\mathcal{L}_n, \mathcal{M}_m) = S_{np}^*(\mathcal{L}_n, \mathcal{M}_n)$ and $C_{\max}\left(S_{np}^*(\mathcal{L}_n, \mathcal{M}_m)\right) = C_{\max}\left(S_{np}^*(\mathcal{L}_n, \mathcal{M}_n)\right)$, while in the preemptive case $C_{\max}\left(S_p^*(\mathcal{L}_n, \mathcal{M}_m)\right) = \max\{T_u | 1 \leq u \leq n < m\} = C_{\max}\left(S_p^*(\mathcal{L}_n, \mathcal{M}_n)\right)$.

Since for an instance $(\mathcal{L}_n, \mathcal{M}_m)$ with $n < m$ the removal of the $m - n$ slowest machines does not change the value of the power of preemption, for the purpose of studying an upper bound on it we only need to consider instances in which there are at least as many jobs as machines.

We focus on a slightly modified version of the LPT algorithm, which can be stated as follows.

Algorithm LPTm

Step 1. If required, renumber the jobs so that the m longest jobs are numbered in accordance with (3), while the other jobs are numbered arbitrarily.

Step 2. At any time that a machine becomes available, take the first job in the current list $\mathcal{L}_n$ and assign it to the machine on which it will complete as early as possible. Remove the assigned job from the list.

Step 3. Repeat Step 2 until all jobs are assigned.

Compared to the full version of the LPT algorithm, the modified Algorithm LPTm requires only $O(m \log m + nm)$ time, since finding and sorting m longest jobs takes $O(m \log m)$ time. From now on, a schedule created by Algorithm LPTm for an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ will be called $S_{LPT}(I)$.

3 Upper Bounds on the Power of Preemption

In this section, we analyze the performance of Algorithm LPTm from the point of view of the power of preemption.

▶ **Definition 2.** For an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$, suppose that in a non-preemptive schedule $S_{np}(I)$ the last completed operation is that of processing job $J_h, 1 \le h \le n$, on machine M_k, $1 \le k \le m$. We call job J_h the *terminal* job and machine M_k the *critical* machine.

The main result of this section is the following statement.

▶ **Theorem 3.** *Given an arbitrary instance $I = (\mathcal{L}_n, \mathcal{M}_m)$, where $n \ge m$, let $S_{LPT}(I)$ be a schedule created by Algorithm LPTm. Then*

$$\frac{C_{\max}(S_{LPT}(I))}{C_{\max}(S_p^*(I))} \le 2 - \frac{1}{m}. \tag{6}$$

Proof. The proof is based on the minimal counterexample technique, often used in worst-case analysis of approximation algorithms. Suppose that the theorem is not true, i. e., there exists an instance $(\mathcal{L}_n, \mathcal{M}_m)$, which we call the minimal counterexample, such that

$$\frac{C_{\max}(S_{LPT}(\mathcal{L}_n, \mathcal{M}_m))}{C_{\max}(S_p^*(\mathcal{L}_n, \mathcal{M}_m))} > 2 - \frac{1}{m} \tag{7}$$

and no job can be removed from the instance without violating the inequality (7).

Suppose that in schedule $S_{LPT}(\mathcal{L}_n, \mathcal{M}_m)$ job J_h is the terminal job and machine M_k the critical machine. If $h < n$ then Algorithm LPTm assigns some jobs J_j with $j > h$ after job J_h and they complete earlier than job J_h. Imagine that these jobs are removed from the instance, so that $\mathcal{L}_h = (p_1, p_2, \ldots, p_h)$ is the corresponding list of the processing times. For the modified instance $(\mathcal{L}_h, \mathcal{M}_m)$, we have

$$C_{\max}(S_{LPT}(\mathcal{L}_h, \mathcal{M}_m)) = C_{\max}(S_{LPT}(\mathcal{L}_n, \mathcal{M}_m));$$
$$C_{\max}(S_p^*(\mathcal{L}_h, \mathcal{M}_m)) \le C_{\max}(S_p^*(\mathcal{L}_n, \mathcal{M}_m)),$$

so that

$$\frac{C_{\max}(S_{LPT}(\mathcal{L}_h, \mathcal{M}_m))}{C_{\max}(S_p^*(\mathcal{L}_h, \mathcal{M}_m))} \ge \frac{C_{\max}(S_{LPT}(\mathcal{L}_n, \mathcal{M}_m))}{C_{\max}(S_p^*(\mathcal{L}_n, \mathcal{M}_m))} > 2 - \frac{1}{m}.$$

Thus, if $h < n$ we deduce that instance $(\mathcal{L}_n, \mathcal{M}_m)$ cannot be the minimal counterexample, and we must have that $h = n$. In other words, for the minimal counterexample $(\mathcal{L}_n, \mathcal{M}_m)$

Algorithm LPTm finds a schedule $S_{LPT}(\mathcal{L}_n, \mathcal{M}_m)$ that is terminated by job J_n. Since $n \geq m$, it follows that

$$p_n \leq \frac{1}{m} p(N). \tag{8}$$

For schedule $S_{LPT}(\mathcal{L}_n, \mathcal{M}_m)$, let N_i denote the set of jobs assigned to machine M_i, $1 \leq i \leq m$. For each machine, find the value G_i such that

$$C_{\max}(S_{LPT}(\mathcal{L}_n, \mathcal{M}_m)) = \frac{p(N_i) + G_i}{s_i},\ 1 \leq i \leq m. \tag{9}$$

Let us call the value G_i the *gap* on machine M_i. We can interpret the gap on some machine as the amount of processing that could be additionally assigned to that machine so that the machine completes at exactly time $C_{\max}(S_{LPT}(\mathcal{L}_n, \mathcal{M}_m))$. Clearly, $G_k = 0$, i. e., there is no gap on the critical machine M_k. Besides, we must have that

$$p_n \geq \max\{G_i | 1 \leq i \leq m,\ i \neq k\}. \tag{10}$$

If the latter inequality had not been true, then Algorithm LPTm would have assigned job J_n to another machine, producing a schedule with a smaller makespan than $C_{\max}(S_{LPT}(\mathcal{L}_n, \mathcal{M}_m))$.

Summing up the equalities (9) we deduce

$$\begin{aligned}\sum_{i=1}^{m} p(N_i) + \sum_{i=1}^{m} G_i &= p(N) + \sum_{i=1}^{m} G_i = C_{\max}(S_{LPT}(\mathcal{L}_n, \mathcal{M}_m)) \sum_{i=1}^{m} s_i \\ &> \left(2 - \frac{1}{m}\right) C_{\max}\left(S_p^*(\mathcal{L}_n, \mathcal{M}_m)\right) S_m,\end{aligned}$$

where the last inequality is due to (7). Since $C_{\max}\left(S_p^*(\mathcal{L}_n, \mathcal{M}_m)\right) \geq T_m = p(N)/S_m$, we deduce $\sum_{i=1}^{m} G_i > \left(1 - \frac{1}{m}\right) p(N)$. On the critical machine the gap is equal to zero, therefore the largest gap on the remaining machines is at least $\frac{1}{m-1} \sum_{i=1}^{m} G_i$. This and (10) yield

$$p_n \geq \frac{1}{m-1} \sum_{i=1}^{m} G_i > \frac{p(N)}{m},$$

which contradicts (8). Thus, the minimal counterexample does not exist and (6) holds. ◀

Notice that Theorem 3 holds for all instances, irrespective of their class. However, below we show that the established upper bound can be reduced for instances $I = (\mathcal{L}_n, \mathcal{M}_m)$ that are known to belong to Class r, $1 \leq r \leq m-1$. If r is not unique, we select the value that is the closest to $m/2$.

For r, $1 \leq r \leq m-1$, define the lists $\mathcal{L}'_r$ and $\mathcal{M}'_r$ obtained from the lists $\mathcal{L}_n$ and $\mathcal{M}_m$ by the removal of the r longest jobs and the r fastest machines, respectively. In other words, $\mathcal{L}'_r = (p_{r+1}, \ldots, p_n)$ and $\mathcal{M}'_r = (s_{r+1}, \ldots, s_m)$. The following algorithm for creating a non-preemptive schedule for an instance I of Class r applies Algorithm LPTm to two instances, $(\mathcal{L}_r, \mathcal{M}_r)$ and $(\mathcal{L}'_r, \mathcal{M}'_r)$.

Algorithm LPTr

Step 1. Given an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ of Class r, $1 \leq r \leq m-1$, split I into two instances $(\mathcal{L}_r, \mathcal{M}_r)$ and $(\mathcal{L}'_r, \mathcal{M}'_r)$.

Step 2. Run Algorithm LPTm twice to find a schedule $S_{LPT}(\mathcal{L}_r, \mathcal{M}_r)$ and a schedule $S_{LPT}(\mathcal{L}'_r, \mathcal{M}'_r)$.

Step 3. Output schedule $S_{LPT(r)}(\mathcal{L}_n, \mathcal{M}_m)$ obtained by combining the schedules $S_{LPT}(\mathcal{L}_r, \mathcal{M}_r)$ and $S_{LPT}(\mathcal{L}'_r, \mathcal{M}'_r)$.

The algorithm requires $O(m \log m + nm)$ time. For its analysis, define

$$T'_r = \frac{\sum_{j=r+1}^{n} p_j}{\sum_{i=r+1}^{m} s_i}.$$

▶ **Lemma 4.** *For an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ of Class r, $1 \le r \le m-1$, the inequality*

$$T_r \ge T'_r \tag{11}$$

holds.

Proof. Since for an For an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ of Class r by definition the inequality $T_r \ge T_m$ holds, we deduce that

$$\begin{aligned} 0 \quad \le \quad T_r - T_m &= \frac{\sum\limits_{j=1}^{r} p_j}{\sum\limits_{i=1}^{r} s_i} - \frac{\sum\limits_{j=1}^{n} p_j}{\sum\limits_{i=1}^{m} s_i} = \frac{\sum\limits_{j=1}^{r} p_j \sum\limits_{i=1}^{m} s_i - \sum\limits_{j=1}^{n} p_j \sum\limits_{i=1}^{r} s_i}{\sum\limits_{i=1}^{r} s_i \sum\limits_{i=1}^{m} s_i} \\ = \quad & \frac{\sum\limits_{j=1}^{r} p_j \left(\sum\limits_{i=1}^{m} s_i - \sum\limits_{i=1}^{r} s_i \right) - \sum\limits_{j=r+1}^{n} p_j \sum\limits_{i=1}^{r} s_i}{\sum\limits_{i=1}^{r} s_i \sum\limits_{i=1}^{m} s_i} = \frac{\sum\limits_{i=1}^{r} p_j \sum\limits_{i=r+1}^{m} s_i - \sum\limits_{j=r+1}^{n} p_j \sum\limits_{i=1}^{r} s_i}{\sum\limits_{i=1}^{r} s_i \sum\limits_{i=1}^{m} s_i}, \end{aligned}$$

which implies that (11) holds. ◀

▶ **Theorem 5.** *Given an arbitrary instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ of Class r, where $n \ge m$ and $1 \le r \le m-1$, let $S_{LPT(r)}(I)$ be a schedule created by Algorithm LPTr. Then*

$$\rho_m = \frac{C_{\max}\left(S^*_{np}(I)\right)}{C_{\max}\left(S^*_{p}(I)\right)} \le \frac{C_{\max}\left(S_{LPT(r)}(I)\right)}{C_{\max}\left(S^*_{p}(I)\right)} \le \max\left\{2 - \frac{1}{r}, 2 - \frac{1}{m-r}\right\}. \tag{12}$$

Proof. Applying Theorem 3 to instances $(\mathcal{L}_r, \mathcal{M}_r)$ and $(\mathcal{L}'_r, \mathcal{M}'_r)$, we obtain

$$\begin{aligned} \frac{C_{\max}\left(S^*_{np}(\mathcal{L}_r, \mathcal{M}_r)\right)}{C_{\max}\left(S^*_{p}(\mathcal{L}_r, \mathcal{M}_r)\right)} \quad &\le \quad \frac{C_{\max}\left(S_{LPT}(\mathcal{L}_r, \mathcal{M}_r)\right)}{C_{\max}\left(S^*_{p}(\mathcal{L}_r, \mathcal{M}_r)\right)} = \frac{C_{\max}\left(S_{LPT}(\mathcal{L}_r, \mathcal{M}_r)\right)}{T_r} \le 2 - \frac{1}{r}; \\ \frac{C_{\max}\left(S^*_{np}(\mathcal{L}'_r, \mathcal{M}'_r)\right)}{C_{\max}\left(S^*_{p}(\mathcal{L}'_r, \mathcal{M}'_r)\right)} \quad &\le \quad \frac{C_{\max}\left(S_{LPT}(\mathcal{L}'_r, \mathcal{M}'_r)\right)}{C_{\max}\left(S^*_{p}(\mathcal{L}'_r, \mathcal{M}'_r)\right)} \le \frac{C_{\max}\left(S_{LPT}(\mathcal{L}'_r, \mathcal{M}'_r)\right)}{T'_r} \le 2 - \frac{1}{m-r}. \end{aligned}$$

Due to (11)

$$\begin{aligned} \frac{C_{\max}\left(S^*_{np}(\mathcal{L}_n, \mathcal{M}_m)\right)}{C_{\max}\left(S^*_{p}(\mathcal{L}_n, \mathcal{M}_m)\right)} \quad &\le \quad \frac{C_{\max}\left(S_{LPT(r)}(\mathcal{L}_n, \mathcal{M}_m)\right)}{T_r} \\ &= \quad \frac{\max\left\{C_{\max}\left(S_{LPT}(\mathcal{L}_r, \mathcal{M}_r)\right), C_{\max}\left(S_{LPT}(\mathcal{L}'_r, \mathcal{M}'_r)\right)\right\}}{T_r} \\ &\le \quad \max\left\{\frac{C_{\max}\left(S_{LPT}(\mathcal{L}_r, \mathcal{M}_r)\right)}{T_r}, \frac{C_{\max}\left(S_{LPT}(\mathcal{L}'_r, \mathcal{M}'_r)\right)}{T'_r}\right\} \\ &\le \quad \max\left\{2 - \frac{1}{r}, 2 - \frac{1}{m-r}\right\}, \end{aligned}$$

as required. ◀

4 Proofs of Tightness

In this section, we prove that the established bounds on the power of preemption are tight.

4.1 Class m Instances

We start with instances of Class m. A tight instance I of this class satisfies the equality

$$\rho_m = \frac{C_{\max}\left(S^*_{np}(I)\right)}{C_{\max}\left(S^*_{p}(I)\right)} = 2 - \frac{1}{m}. \tag{13}$$

We exhibit the instances for which (13) holds; moreover, we describe the necessary and sufficient conditions for an instance of Class m to be tight. Let us introduce a special class of instances of the problem that plays a crucial role in establishing tightness of the bounds on the power of preemption.

▶ **Definition 6.** For the problem with m uniform machines, an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ is called *canonical* if for each machine M_k there exists an optimal non-preemptive schedule such that M_k is the only critical machine.

Under the usual assumption that $n \geq m$, let $\mathcal{I}$ be a set of instances $I = (\mathcal{L}_n, \mathcal{M}_m)$ such that

- The processing times satisfy $p_j = p,\ j \in N$;
- The speeds are positive integers that for a positive W satisfy

$$s_1 \geq s_2 \geq \cdots \geq s_m;\ 1 \leq W s_i \leq m;\ \sum_{i=1}^{m} s_i = \frac{n+m-1}{W}.$$

▶ **Lemma 7.** *For any instance $I = (\mathcal{L}_n, \mathcal{M}_m) \in \mathcal{I}$ the equality*

$$\frac{C_{\max}\left(S^*_{np}\left(I\right)\right)}{C_{\max}(S^*_{p}\left(I\right))} = 1 + \frac{m-1}{n}$$

holds.

Proof. For an optimal non-preemptive schedule $S^*_{np}(I)$, let n_i denote the number of jobs assigned to machine M_i. If $n_i \leq W s_i - 1$, $1 \leq i \leq m$, then we derive a contradiction:

$$n = \sum_{i=1}^{m} n_i \leq W \sum_{i=1}^{m} s_i - m = (n+m-1) - m = n-1.$$

Thus, in $S^*_{np}(I)$ at least one machine should get $n_i \geq W s_i$ jobs, i. e., $C_{\max}\left(S^*_{np}(I)\right) \geq Wp$. The smallest value of the makespan is achieved if for an arbitrary k, $1 \leq k \leq m$, machine M_k gets exactly $n_k = W s_k$ jobs, so that $C_{\max}\left(S^*_{np}(I)\right) = Wp$, which is the completion time of the last job assigned to machine M_k. To make sure that all other machines complete earlier than time p, assign exactly $n_i = W s_i - 1$ jobs to machine M_i, $1 \leq i \leq m,\ i \neq k$. This allocation is feasible, i. e., all n jobs are distributed, since

$$n = \sum_{i=1}^{m} n_i = W \sum_{i=1}^{m} s_i - (m-1) = n.$$

Thus, we derive that for any instance $I \in \mathcal{I}$ the equality

$$\frac{C_{\max}\left(S^*_{np}\left(I\right)\right)}{C_{\max}(S^*_{p}\left(I\right))} = \frac{Wp}{\frac{Wpn}{n+m-1}} = 1 + \frac{m-1}{n},$$

holds, i. e., I is a tight instance. ◀

The lemma below states that set $\mathcal{I}$ consists of instances of Class m.

▶ **Lemma 8.** *Any instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ such that $p_1 = \cdots = p_n = p$ belongs to Class m.*

Proof. For any u, $1 \leq u \leq m-2$, we have that $T_u = up/S_u$, so that

$$T_u - T_{u+1} = \frac{up}{S_u} - \frac{(u+1)\,p}{S_u + s_{u+1}} = \frac{us_{u+1} - S_u}{S_u\,(S_u + s_{u+1})}p.$$

Since

$$S_u = \sum_{i=1}^{u} s_i \geq us_u \geq us_{u+1},$$

we deduce that the sequence $T_1, T_2, \ldots T_{m-1}$ is non-decreasing. Besides,

$$\begin{aligned} T_{m-1} - T_m &= \frac{(m-1)\,p}{S_{m-1}} - \frac{np}{S_{m-1} + s_m} \\ &= \frac{(m-1)\,s_m - (n+1-m)\,S_{m-1}}{S_{m-1}\,(S_{m-1} + s_m)}p \leq 0. \end{aligned}$$

This proves the lemma. ◀

Under the assumption that $n \geq m$, the value $1 + (m-1)/n$ reaches its maximum of $2 - 1/m$ if $n = m$. Combining Theorem 3, Lemma 7 and Lemma 8, we derive the following statement.

▶ **Corollary 9.** *For instances of Class m the power of preemption is $2 - \frac{1}{m}$, and this value cannot be reduced for instances of this class.*

As far as the set $\mathcal{I}$ is concerned, a stronger statement can be proved.

▶ **Theorem 10.** *For an instance $I = (\mathcal{L}_n, \mathcal{M}_m)$ of Class m to be tight, it is necessary and sufficient that I is an instance of set $\mathcal{I}$ with $n = m$.*

Proof. Sufficiency of the theorem immediately follows from Lemma 7. To prove necessity, first notice that it follows from the tightness of instance I that is does not belong to Class r for any r, $1 \leq r \leq m-1$. Due to Theorem 3 we have that

$$2 - \frac{1}{m} = \frac{C_{\max}\left(S^*_{np}(I)\right)}{C_{\max}\left(S^*_p(I)\right)} \leq \frac{C_{\max}\left(S_{LPT}(I)\right)}{C_{\max}\left(S^*_p(I)\right)} \leq 2 - \frac{1}{m}. \tag{14}$$

This implies that in (14) all inequalities hold as equalities, i. e., for a tight instance I Algorithm LPTm in fact finds an optimal non-preemptive schedule. In the remainder of this proof we can deal with schedule $S_{LPT}(I)$ instead of schedule $S^*_{np}(I)$.

If in schedule $S_{LPT}(I)$ some job J_h with $h < n$ is terminal, then the jobs $h+1, \ldots, n$ can be removed from the instance. Since I is a Class m instance and does not belong to any other class, the removal of the jobs reduces the makespan of the optimal preemptive schedule, i. e., for the modified instance I', $C_{\max}\left(S^*_p(I')\right) < C_{\max}\left(S^*_p(I)\right)$. On the other hand we have $C_{\max}\left(S_{LPT}(I')\right) = C_{\max}\left(S_{LPT}(I)\right)$, so that

$$\frac{C_{\max}\left(S_{LPT}(I')\right)}{C_{\max}\left(S^*_p(I')\right)} > \frac{C_{\max}\left(S_{LPT}(I)\right)}{C_{\max}\left(S^*_p(I)\right)} = \frac{C_{\max}\left(S^*_{np}(I)\right)}{C_{\max}\left(S^*_p(I)\right)} = 2 - \frac{1}{m}.$$

However, this implies Theorem 3 does not hold for instance I'. Thus, in what follows we assume that in $S_{LPT}(I)$ the terminal job is job J_n and hence unique. For job J_n (8) holds due to $n \geq m$.

Similarly to the proof of Theorem 3, for schedule $S_{LPT}(I)$ let G_i be the gap on machine M_i that is defined by (9). The gap analysis of schedule $S_{LPT}(I)$ leads to

$$p(N) + \sum_{i=1}^{m} G_i = C_{\max}\left(S_{LPT}\left(I\right)\right) \sum_{i=1}^{m} s_i = \left(2 - \frac{1}{m}\right) C_{\max}\left(S_p^*\left(I\right)\right) S_m.$$

Since I is a Class m instance, we deduce from $C_{\max}\left(S_p^*\left(I\right)\right) = p\left(N\right)/S_m$ that

$$\sum_{i=1}^{m} G_i = \left(\frac{m-1}{m}\right) p(N).$$

Since in schedule $S_{LPT}\left(I\right)$ the gap on the critical machine is zero, it follows that the largest gap $G_{\max} = \max\left\{G_i | 1 \le i \le m\right\}$ is no smaller than $p\left(N\right)/m$. On the other hand, $G_{\max}$ does not exceed p_n; otherwise Algorithm LPTm would have assigned job J_n to the machine with the largest gap. Combining this with (8), we obtain

$$\frac{p\left(N\right)}{m} \le G_{\max} \le p_n \le \frac{p\left(N\right)}{m}.$$

It follows that for $n \ge m$ in the expression above all inequalities hold as strict equalities, and we deduce that

- $m = n$, i. e., in a tight instance I, the number of jobs is equal to the number of machines.
- in a tight instance I all processing times are equal, i. e., $p_j = p,\ j \in N$, where $p = p(N)/n$.
- the largest gap $G_{\max}$ is equal to p.

Lemma 8 confirms that a tight instance I belongs to Class m, i. e., $C_{\max}\left(S_p^*\left(I\right)\right) = mp/S_m$. The total gap on all machines is equal to

$$\sum_{i=1}^{m} G_i = \left(m-1\right) p. \tag{15}$$

In schedule $S_{LPT}\left(I\right)$ the terminal job is unique, i. e., there are $m-1$ non-critical machines, each with a non-zero gap. Since the largest gap is p, it follows from (15) that in schedule $S_{LPT}\left(I\right)$ at the time of assigning the last job J_n the gaps on all machines are the same and equal to p. This means that any machine can be made critical, while the remaining machines will complete earlier. In other words, I is a canonical instance, and for each machine M_i, there exists an optimal non-preemptive schedule in which machine M_i is critical.

For i, $1 \le i \le m$, let k_i denote the number of jobs on M_i in an optimal schedule in which machine M_i is critical, i. e., $C_{\max}\left(S_{np}^*\left(I\right)\right) = k_i p/s_i,\ i = 1, \ldots, m$. We deduce

$$C_{\max}\left(S_{np}^*\left(I\right)\right) \sum_{i=1}^{m} s_i = p \sum_{i=1}^{m} k_i.$$

On the other hand, since I is a Class m instance, the equalities $C_{\max}\left(S_p^*\left(I\right)\right) = p\left(N\right)/\sum_{i=1}^{m} s_i = mp/\sum_{i=1}^{m} s_i$ hold, and we derive

$$C_{\max}\left(S_{np}^*\left(I\right)\right) \sum_{i=1}^{m} s_i = \left(2 - \frac{1}{m}\right) C_{\max}\left(S_p^*\left(I\right)\right) \sum_{i=1}^{m} s_i = \left(2 - \frac{1}{m}\right) pm.$$

This yields $\sum_{i=1}^{m} k_i = 2m - 1$. Notice that all ratios k_i/s_i, $1 \le i \le m$, are equal. Let W be the value $W = \frac{k_1}{s_1} = \frac{k_2}{s_2} = \ldots \frac{k_m}{s_m}$. Then

$$W \sum_{i=1}^{m} s_i = 2m - 1.$$

and we conclude that I is an instance of set $\mathcal{I}$ with $n = m$. ◀

4.2 Instances of Other Classes

We now demonstrate that for the instances of Class r, $1 \leq r \leq m-1$, the bounds on the power of preemption established in Theorem 5 are tight. Our consideration is split into two cases that depend on the sign of the difference $2r - m$.

▶ **Lemma 11.** *For $n \geq m$, and r such that $1 \leq r \leq m-1$ and $2r \geq m$, there exists an instance $(\mathcal{L}_n, \mathcal{M}_m)$ of Class r such that*

$$\frac{C_{\max}\left(S^*_{np}\left(\mathcal{L}_n, \mathcal{M}_m\right)\right)}{C_{\max}\left(S^*_{p}\left(\mathcal{L}_n, \mathcal{M}_m\right)\right)} = 2 - \frac{1}{r}. \tag{16}$$

Proof. For a given m, take an arbitrary r, such that $1 \leq r \leq m-1$ and $2r \geq m$. To prove the lemma we exhibit an instance $I = (\mathcal{L}_m, \mathcal{M}_m)$ of Class r with m machines and $n = m$ jobs. The $r-1$ faster machines each have speed 2, while all remaining machines have unit speed, i. e., $s_i = 2,\ 1 \leq i \leq r-1$, and $s_i = 1,\ r \leq i \leq m$. The processing times are defined by $p_j = 1,\ 1 \leq j \leq r$, and $p_j = \frac{r}{2r-1} < 1,\ r+1 \leq j \leq m$. We have that

$$T_i = \frac{1}{2},\ 1 \leq i \leq r-1;\ T_r = \frac{r}{2r-1} > \frac{1}{2};\ T_i = \frac{r + (i-r)\,p_i}{2r - 1 + (i-r)} = \frac{r}{2r-1},\ r+1 \leq i \leq m.$$

Here, $T_r = T_{r+1} = \cdots = T_m$ and r is the index that is closest to $m/2$ due to $r \geq m/2$. Thus, instance I belongs to Class r and and $C_{\max}\left(S^*_p\left(I\right)\right) = r/\left(2r-1\right)$.

On the other hand, it can be verified that $C_{\max}\left(S^*_{np}\left(I\right)\right) = 1$. Indeed, had $C_{\max}\left(S^*_{np}\left(I\right)\right)$ been strictly less than 1 then each of the faster machines of speed 2 should have processed exactly one job of unit duration, and therefore the remaining job of unit duration would have been assigned to a machine of unit speed, a contradiction. Thus, $C_{\max}\left(S^*_{np}\left(I\right)\right)/C_{\max}\left(S^*_{p}\left(I\right)\right) = 2 - 1/r$, so that (16) holds. ◀

▶ **Lemma 12.** *For $n \geq m$, and r such that $1 \leq r \leq m-1$ and $2r < m$, there exists an instance $(\mathcal{L}_n, \mathcal{M}_m)$ of Class r such that*

$$\frac{C_{\max}\left(S^*_{np}\left(\mathcal{L}_n, \mathcal{M}_m\right)\right)}{C_{\max}\left(S^*_{p}\left(\mathcal{L}_n, \mathcal{M}_m\right)\right)} = 2 - \frac{1}{m-r}. \tag{17}$$

Proof. For a given m, take an arbitrary r, such that $1 \leq r \leq m-1$ and $2r < m$. To prove the lemma we exhibit an instance $I = (\mathcal{L}_m, \mathcal{M}_m)$ of Class r with m machines and $n = m$ jobs. The speeds of all machines are equal to 2, except machine M_m, which has unit speed, i. e., $s_i = 2,\ 1 \leq i \leq m-1$, and $s_m = 1$. Compute

$$Q = \frac{m-r}{2\left(m-r\right)-1}$$

and define the processing times as $p_j = 2Q > 1,\ 1 \leq j \leq r$, and $p_j = 1,\ r+1 \leq j \leq m$. We have that

$$T_i = Q,\ 1 \leq i \leq r;\ T_i = \frac{2rQ + (i-r)}{2r + 2\left(i-r\right)} < Q,\ r+1 \leq i < m;\ T_m = \frac{2rQ + (m-r)}{2m-1} = Q.$$

Here, $T_1 = \cdots = T_r$ and r is the index that is closest to $m/2$ due to $r < m/2$. Thus, instance I belongs to Class r and and $C_{\max}\left(S^*_p\left(I\right)\right) = Q$.

In an optimal non-preemptive schedule a longer job of duration $2Q$ and any other job cannot be completed before time 1 on any machine, since $2Q + 1 > 2$. Thus, in any optimal schedule there are r faster machines of speed 2 each processing exactly one longer job of

duration $2Q$ and completing at time Q. If the slow machine M_m is assigned a job, then it completes it at time 1; otherwise, there exists a faster machine of speed 2 that processes at least two shorter jobs. Thus, $C_{\max}\left(S^*_{np}(I)\right) = 1$, and $C_{\max}\left(S^*_{np}(I)\right)/C_{\max}\left(S^*_{p}(I)\right) = 2 - 1/(m-r)$, so that (17) holds. ◀

Thus, for instances of Class r the bound $2 - \min\{1/r, 1/(m-r)\}$ on the power of preemption is tight.

References

1 O. Braun and G. Schmidt. Parallel processor scheduling with limited number of preemptions. *SIAM Journal on Computing*, 32:671–680, 2003.

2 P. Brucker. *Scheduling Algorithms*, 5th edition. Springer, Berlin, 2007.

3 B. Chen. Parallel machine scheduling for early completion. In J. Y.-T. Leung, ed. *Handbook of Scheduling: Algorithms, Models and Performance Analysis*, Chapman & Hall/CRC, London, pages 9-175–9-184, 2004.

4 J. R. Correa, M. Skutella and J. Verschae. The power of preemption on unrelated machines and applications to scheduling orders. *Mathematics of Operations Research*, 37:379–398, 2012.

5 T. Ebenlendr and J. Sgall. Optimal and online preemptive scheduling on uniformly related machines. *Journal of Scheduling*, 12:517–527, 2009.

6 T. F. Gonzalez and S. Sahni. Preemptive scheduling of uniform processor systems. *Journal of ACM*, 25:92–101, 1978.

7 Y. Jiang, Z. Weng and J. Hu. Algorithms with limited number of preemptions for scheduling on parallel machines. *Journal of Combinatorial Optimization*, 27:711–723, 2014.

8 A. Kovács. New approximation bounds for LPT scheduling. *Algorithmica*, 57:413–433, 2010.

9 C.-Y. Lee and V. A. Strusevich. Two-machine shop scheduling with an uncapacitated interstage transporter. *IIE Transactions*, 37:725–736, 2005.

10 J.-H. Lin and J. S. Vitter. $\varepsilon-$approximations with minimum packing constraint violation. In: STOC'92 Proceedings of the 24th Annual ACM Symposium on Theory of Computing, ACM: New York, pages 771–782, 1992.

11 K. Rustogi and V. A. Strusevich. Parallel machine scheduling: Impact of adding extra machines. *Operations Research,* 61:1243–1257, 2013.

12 A. J. Soper and V. A. Strusevich. Single parameter analysis of power of preemption on two and three uniform machines. *Discrete Optimization,* 12:26–46, 2014.

13 G. J. Woeginger. A comment on scheduling on uniform machines under chain-like precedence constraints. *Operations Research Letters*, 26:107–109, 2000.

Improved Approximation Algorithms for Matroid and Knapsack Median Problems and Applications*

Chaitanya Swamy

Combinatorics and Optimization, University Waterloo, Waterloo, ON N2L 3G1
cswamy@math.uwaterloo.ca

Abstract

We consider the *matroid median* problem [11], wherein we are given a set of facilities with opening costs and a matroid on the facility-set, and clients with demands and connection costs, and we seek to open an independent set of facilities and assign clients to open facilities so as to minimize the sum of the facility-opening and client-connection costs. We give a simple 8-approximation algorithm for this problem based on LP-rounding, which improves upon the 16-approximation in [11]. We illustrate the power and versatility of our techniques by deriving: (a) an 8-approximation for the *two-matroid median* problem, a generalization of matroid median that we introduce involving two matroids; and (b) a 24-approximation algorithm for *matroid median with penalties*, which is a vast improvement over the 360-approximation obtained in [11]. We show that a variety of seemingly disparate facility-location problems considered in the literature—data placement problem, mobile facility location, k-median forest, metric uniform minimum-latency UFL—in fact reduce to the matroid median or two-matroid median problems, and thus obtain *improved* approximation guarantees for all these problems. Our techniques also yield an improvement for the knapsack median problem.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.2 Discrete Mathematics

Keywords and phrases Approximation algorithms, LP rounding, facility location, matroid and submodular polyhedra, knapsack constraints

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.403

1 Introduction

We investigate facility location problems wherein the set of open facilities have to satisfy some matroid independence constraints or knapsack constraints. Specifically, we consider the *matroid median problem*, which is defined as follows. As in the uncapacitated facility location problem, we are given a set of facilities $\mathcal{F}$ and a set of clients $\mathcal{D}$. Each facility i has an *opening cost* of f_i. Each client $j \in \mathcal{D}$ has demand d_j and assigning client j to facility i incurs an *assignment cost* of $d_j c_{ij}$ proportional to the distance between i and j. Further, we are given a matroid $M = (\mathcal{F}, \mathcal{I})$ on the set of facilities. The goal is to choose a set $F \in \mathcal{I}$ of facilities to open that forms an independent set in M, and assign each client j to a facility $i(j) \in F$ so as to minimize the total facility-opening and client-assignment costs, that is, $\sum_{i \in F} f_i + \sum_{j \in \mathcal{D}} d_j c_{i(j)j}$. We assume that the facilities and clients are located in a common metric space, so the distances c_{ij} form a metric.

* A full version is available on the CS arXiv, see http://arxiv.org/abs/1310.7834.
This work was supported in part by NSERC grant 327620-09, an NSERC Discovery Accelerator Supplement Award, and an Ontario Early Researcher Award.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 403–418

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

The matroid median problem generalizes the metric *k-median* problem, which is the special case where M is a uniform matroid (and there are no facility-opening costs), and is thus, *NP*-hard. The matroid median problem without facility-opening costs was introduced recently by Krishnaswamy et al. [11], who gave a 16-approximation algorithm for this problem.

Our contributions are threefold.

- We devise an improved 8-approximation algorithm for the matroid-median problem (Section 3). Moreover, notably, our algorithm is significantly simpler and cleaner than the one in [11], and satisfies the stronger property that it is a *Lagrangian-multiplier-preserving* 8-approximation algorithm (see Remark 3.4). The effectiveness and versatility of our simpler approach for matroid median is further highlighted when we consider some natural extensions of matroid median in Section 4. We leverage the techniques underlying our simpler and cleaner algorithm for matroid median to devise: (a) an 8-approximation algorithm for the *two-matroid median* problem (Section 4.1), which is an extension that we introduce involving two matroids that captures some interesting facility-location problems considered in the literature; and (b) a 24-approximation algorithm (Section 4.2) for the *matroid median problem with penalties*, wherein we are allowed to leave client unassigned and incur a penalty for each unassigned client; this constitutes a vast improvement over the approximation ratio of 360 obtained by Krishnaswamy et al. [11].
- We show that the matroid median and two-matroid median problem turn out to be rather fundamental problems by showing in Section 5 that a variety of facility location problems that have been considered in the literature can be cast as instances of matroid median or two-matroid median. These include the data placement problem [2, 3], mobile facility location [9, 1], k-median forest [10], and metric uniform minimum-latency UFL [4]. This not only gives a unified framework for viewing these seemingly disparate problems, but also our approximation guarantee of 8 *yields improved, and in some cases, the first, approximation guarantees for all these problems.*
- We adapt our techniques to also obtain an improvement for the knapsack median problem [11, 12] (Section 6).

Our improvement for matroid median comes from an improved, simpler rounding procedure for a natural LP relaxation of the problem also considered in [11]. We show that a clustering step introduced in [5] for the k-median problem coupled with two applications of the integrality of the intersection of two submodular (or matroid) polyhedra—one to obtain a half-integral solution, and another to obtain an integral solution—suffices to obtain the desired approximation ratio. In contrast, the algorithm in [11] starts off with the clustering step in [5], but then further dovetails the rounding procedure of [5] creating trees, then stars, and then applies the integrality of the intersection of two submodular polyhedra.

There is great deal of similarity between the the rounding algorithm of [11] for matroid median and the rounding algorithm of Baev and Rajaraman [2] for the data placement problem, who also perform the initial clustering step in [5] and then create trees and then stars and use these to obtain an integral solution. In contrast, our simpler, improved rounding algorithm is similar to the rounding algorithm in [3] for data placement, who use the initial clustering step of [5] coupled with two min-cost flow computations—one to obtain a half-integral solution and another to obtain an integral solution—to obtain the final solution. These similarities are not surprising since, as mentioned above, we show in Section 5 that the data-placement problem is a special case of the matroid median problem. In fact, our improvements are analogous to those obtained for the data-placement problem by Baev, Rajaraman, and Swamy [3] over the guarantees in [2], and stem from similar insights.

A common theme to emerge from our work and [3] is that in various settings, the initial

clustering step introduced by [5] imparts sufficient structure to the fractional solution so that one can then round it using two applications of suitable integrality-results from combinatorial optimization. First, this initial clustering can be used to derive a half-integral solution. This was observed explicitly in [2] and is implicit in [11], and making this explicit yields significant dividends. Second, and this is the oft-overlooked insight (in [2, 11]), a half-integral solution can be easily rounded, and in a better way, *without resorting to creating trees and then stars etc. as in the algorithm of [5].* This is due to the fact that a half-integral solution is already "filtered": if client j is assigned to facility i fractionally, then one can bound c_{ij} in terms of the assignment cost paid by the fractional solution for j (see Section 3). This enables one to use a standard facility-location clustering step to set up a suitable combinatorial-optimization problem possessing an integrality property, and hence, round the half-integral solution. The resulting algorithm is typically both simpler and has a better approximation ratio than what one would obtain by mimicking the steps of [5] involving creating trees, stars etc.

Recently, Charikar and Li [6] obtained a 9-approximation algorithm for the matroid-median problem; our results were obtained independently. While there is some similarity between our ideas and those in [6], we feel that our algorithm and analysis provides a more illuminating explanation of why matroid median and some of its extensions (e.g., two-matroid median, matroid median with penalties; see Section 4) are "easy" to approximate, whereas other variants such as matroid-intersection median (Section 4) are inapproximable. It is possible that our ideas coupled with the dependent-rounding procedure used in [6] for the k-median problem may lead to further improvements for the matroid median problem; we leave this as future work.

2 An LP Relaxation for Matroid Median

We can express the matroid median problem as an integer program and relax the integrality constraints to get an LP. Throughout we use i to index facilities in $\mathcal{F}$, and j to index clients in $\mathcal{D}$. Let r denote the rank function of the matroid $M = (\mathcal{F}, \mathcal{I})$.

$$\begin{aligned} \min \quad & \sum_i f_i y_i + \sum_j \sum_i d_j c_{ij} x_{ij} && && \text{(P)} \\ \text{s.t.} \quad & \sum_i x_{ij} = 1 && \forall j \\ & \sum_{i \in S} y_i \leq r(S) && \forall S \subseteq \mathcal{F} \\ & 0 \leq x_{ij} \leq y_i && \forall i, j. \end{aligned}$$

Variable y_i indicates if facility i is open, and x_{ij} indicates if client j is assigned to facility i. The first and second constraints say that each client must be assigned to an open facility. The third constraint encodes the matroid independence constraint. An integer solution corresponds exactly to a solution to our problem. We note that (P) can be solved in polytime since (for example) a polytime algorithm for submodular-function minimization yields an efficient separation oracle.

3 A Simple 8-Approximation Algorithm via LP-Rounding

Let (x, y) denote an optimal solution to (P) and OPT be its value. We first describe a simple algorithm to round (x, y) to an integer solution losing a factor of at most 10. In Section 3.4, we use some additional insights to improve the approximation ratio to 8. We use the terms connection cost and assignment cost interchangeably.

3.1 Overview of the Algorithm

We first give a high level description of the algorithm. Suppose for a moment that the optimal solution (x, y) satisfies the following property:

for every facility i, there is *at most one* client j such that $x_{ij} > 0$. $(*)$

Let $\mathcal{F}_j = \{i : x_{ij} > 0\}$. Notice that the $\mathcal{F}_j$ sets are disjoint. We may assume that for $i \in \mathcal{F}_j$, we have $y_i = x_{ij}$, so the objective function is a linear function of only the y_i variables. We can then set up the following matroid intersection problem. The first matroid is M restricted to $\bigcup_j \mathcal{F}_j$. The second matroid M' (on the same ground set $\bigcup_j \mathcal{F}_j$) is the partition matroid defined by the $\mathcal{F}_j$ sets; that is, a set is independent in M' if it contains at most one facility from each $\mathcal{F}_j$. Notice the y_i-variables yield a fractional point in the *intersection of the matroid polyhedron of M and the matroid-base polyhedron of M'*. Since the intersection of these two polyhedra is known to be integral (see, e.g., [8]), this means that we can round (x, y) to an integer solution of no greater cost. Of course, the LP solution need not have property $(*)$ so our goal will be to transform (x, y) to a solution that has this property without increasing the cost by much.

Roughly speaking we want to do the following: cluster the clients in $\mathcal{D}$ around certain 'centers' (also clients) such that (a) every client k is assigned to a "nearby" cluster center j whose LP assignment cost is less than that of k, and (b) the facilities serving the cluster centers in the fractional solution (x, y) are disjoint. So, the modified instance where the demand of a client is moved to the center of its cluster has a fractional solution, namely the solution induced by (x, y), that satisfies $(*)$ and has cost at most OPT. Furthermore, given a solution to the modified instance we can obtain a solution to the original instance losing a small additive factor. One option is to use the decomposition method of Shmoys et al. [13] for uncapacitated facility location (UFL) that produces precisely such a clustering. The problem however is that [13] uses filtering which involves blowing up the x_{ij} and y_i values, thus violating the matroid-rank packing constraints. Chudak and Shmoys [7] use the same clustering idea but without filtering, using the dual solution to bound the cost. The difficulty here with this approach is that there are terms with negative coefficients in the dual objective function that correspond to the primal matroid-rank constraints. Although [14] showed that it is possible to overcome this difficulty in certain cases, the situation here looks more complicated and it is not clear how to use their techniques.

Instead, we use the clustering technique of Charikar et al. [5] to cluster clients and first obtain a *half-integral solution* $(\hat{x}, \hat{y})$, that is, every $\hat{x}_{ij}, \hat{y}_i \in \left\{0, \frac{1}{2}, 1\right\}$, to the modified instance with cluster centers, losing a factor of 3. Further, any solution here will give a solution to the original instance while increasing the cost by at most $4 \cdot OPT$. Now we use the clustering method of [13] *without any filtering*, since the half-integral solution $(\hat{x}, \hat{y})$ is essentially already filtered; if client j is assigned to i and i' in $\hat{x}$, then $c_{ij}, c_{i'j} \leq 2(c_{ij}\hat{x}_{ij} + c_{i'j}\hat{x}_{i'j})$. This final step causes us to lose an additive factor equal to the cost of $(\hat{x}, \hat{y})$, so overall we get an approximation ratio of $4 + 3 + 3 = 10$. In Section 3.4, we show that by further exploiting the structure of the half-integral solution, we can give a better bound on the cost of the integer solution and thus obtain an 8-approximation.

We now describe each of these steps in detail; omitted proofs appear in the full version. Let $\bar{C}_j = \sum_i c_{ij}x_{ij}$ denote the cost incurred by the LP solution to assign one unit of demand of client j. Given a vector $v \in \mathbb{R}^{\mathcal{F}}$ and a set $S \subseteq \mathcal{F}$, we use $v(S)$ to denote $\sum_{i \in S} v_i$.

3.2 Obtaining a Half-integral Solution $(\hat{x}, \hat{y})$

Step I: Consolidating Demands around Centers. We first consolidate (or cluster) the demand of clients at certain clients, that we call *cluster centers*. We do not modify the fractional solution (x, y) but only modify the demands so that for some clients j, the demand d_j is "moved" to a "nearby" center k. We assume every client has non-zero demand (we can simply get rid of zero-demand clients).

Set $d'_j \leftarrow 0$ for every j. Consider the clients in increasing order of $\bar{C}_j$. For each client j, if there exists a client k such that $d'_k > 0$ and $c_{jk} \leq 4\max(\bar{C}_j, \bar{C}_k) = 4\bar{C}_k$, set $d'_k \leftarrow d'_k + d_j$, otherwise set $d'_j \leftarrow d_j$. Let $D = \{j \in \mathcal{D} : d'_j > 0\}$. Each client in D is a cluster center. Let $OPT' = \sum_i f_i y_i + \sum_{j \in D, i} d'_j c_{ij} x_{ij}$ denote the cost of (x, y) for the modified instance consisting of the cluster centers.

▶ **Lemma 3.1.** *(i) If $j, k \in D$, then $c_{jk} \geq 4\max(\bar{C}_j, \bar{C}_k)$, (ii) $OPT' \leq OPT$, and (iii) any solution (x', y') to the modified instance can be converted to a solution to the original instance incurring an additional cost of at most $4 \cdot OPT$.*

From now on we focus on the modified instance with client set D and modified demands d'_j. At the very end we will use the above lemma to translate an integer solution to the modified instance to an integer solution to the original instance.

Step II: Transforming to a Half-integral Solution. We define the cluster of a client $j \in D$ to be the set F_j of all facilities i such that j is the center in D closest to i, that is, $F_j = \{i : c_{ij} = \min_{k \in D} c_{ik}\}$, with ties broken arbitrarily. Let $F'_j \subseteq F_j = \{i \in F_j : c_{ij} \leq 2\bar{C}_j\}$. Clearly the sets F_j for $j \in D$ are disjoint. By property (i) of Lemma 3.1, we have that F_j contains all the facilities i such that $c_{ij} \leq 2\bar{C}_j$. So $\sum_{i \in F'_j} x_{ij} = \sum_{i : c_{ij} \leq 2\bar{C}_j} x_{ij} \geq \frac{1}{2}$ by Markov's inequality.

To obtain the half-integral solution, we define a suitable vector y' that lies in a polytope with half-integral extreme points and construct a linear function $T(.)$ such that $T(y')$ bounds the cost of a fractional solution. We show that $T(y') \leq 3 \cdot OPT'$. This implies that one can obtain a "better" half-integral vector $\hat{y}$, which we then argue yields a half-integral solution $(\hat{x}, \hat{y})$ to the modified instance of cost at most $T(\hat{y}) \leq T(y')$.

Define $\gamma_j := \min_{i \notin F_j} c_{ij}$, and let $G_j = \{i \in F_j : c_{ij} \leq \gamma_j\}$. Note that $\gamma_j \geq 2\bar{C}_j$, so $F'_j \subseteq G_j$. Set $y'_i = x_{ij} \leq y_i$ if $i \in G_j$, and $y'_i = 0$ otherwise. Clearly, $y'(F_j) = y'(G_j) \leq 1$. Then y' lies in the following polytope

$$\mathcal{P} := \Big\{ v \in \mathbb{R}^{\mathcal{F}}_+ : v(S) \leq r(S) \quad \forall S \subseteq \mathcal{F}, \qquad v(F'_j) \geq \tfrac{1}{2}, \quad v(G_j) \leq 1 \quad \forall j \in D \Big\}. \tag{1}$$

We claim that $\mathcal{P}$ has half-integral extreme points. The easiest way to see this is to note that any extreme point of $\mathcal{P}$ is defined by a linearly independent system of tight constraints comprising some $v(S) = r(S)$ equalities corresponding to a laminar set system, and some $v(F'_j) = \frac{1}{2}$ and $v(G_j) = 1$ equalities. The constraint matrix of this system thus corresponds to equations coming from two laminar set systems; such a matrix is known to be totally unimodular, and hence the vector v satisfying this system must be a half-integral solution. (The full version also gives a proof based on the integrality of the intersection of two submodular polyhedra.)

Given $v \in \mathbb{R}^{\mathcal{F}}_+$, define $T(v) = \sum_i f_i v_i + \sum_j d'_j \big(\sum_{i \in G_j} c_{ij} v_i + 3\gamma_j (1 - \sum_{i \in G_j} v_i)\big)$. Since $y' \in \mathcal{P}$, this implies that we can obtain a half-integral solution $\hat{y}$ such that $T(\hat{y}) \leq T(y')$. Observe that there is at least one facility $i \in F'_j$ with $\hat{y}_i > 0$; we call the facility $i \in F'_j$ nearest to j the *primary facility* of j and set $\hat{x}_{ij} = \hat{y}_i$. Note that every every client in D

has a *distinct* primary facility. If $\hat{y}_i < 1$, then let i' be the facility nearest to j other than i such that $\hat{y}_{i'} > 0$; we call i' the *secondary facility* of j, and set $\hat{x}_{i'j} = 1 - \hat{x}_{ij}$. Define $\hat{C}_j = \sum_i c_{ij}\hat{x}_{ij}$ and $S_j = \{i : \hat{x}_{ij} > 0\}$.

▶ **Lemma 3.2.** *The cost of* $(\hat{x}, \hat{y})$ *is at most* $3 \cdot OPT' \leq 3 \cdot OPT$.

3.3 Converting $(\hat{x}, \hat{y})$ to an Integer Solution

Step III: Clustering. We cluster the clients in D as follows: pick $j \in D$ with smallest $\hat{C}_j$. Remove every client $k \in D$ such that $S_j \cap S_k \neq \emptyset$; we call j the *center* of k and denote it by $\mathsf{ctr}(k)$. Recurse on the remaining set of clients until no client in D is left. Let D' be the set of clients picked—these are the *new cluster centers*. Note that $\mathsf{ctr}(j) = j$ for every $j \in D'$.

Step IV: The Matroid Intersection Problem. For convenience, we will say that every client $j \in D$ has both a primary facility $i_1(j)$ and a secondary facility $i_2(j)$ with $\hat{x}_{i_1(j)j} = \hat{x}_{i_2(j)j} = \frac{1}{2}$, with the understanding that if j does not have a secondary facility then $i_1(j) = i_1(j)$, and so $\hat{x}_{i_1(j)j} = 1$. Then we have $\hat{C}_j = \frac{1}{2}(c_{i_1(j)j} + c_{i_2(j)j})$ and $c_{i_1(j)j} \leq \hat{C}_j \leq c_{i_2(j)j} \leq 2\hat{C}_j$.

For $i \in \mathcal{F}$, define $\hat{y}'_i = \hat{x}_{ij} \leq \hat{y}_i$ if $i \in S_j$ where $j \in D'$, and $\hat{y}'_i = \hat{y}_i$ otherwise. Then $\hat{y}'$ lies in the polytope

$$\mathcal{R} := \left\{ z \in \mathbb{R}_+^{\mathcal{F}} : z(S) \leq r(S) \quad \forall S \subseteq \mathcal{F}, \qquad z(S_j) = 1 \quad \forall j \in D' \right\}. \tag{2}$$

Observe that $\mathcal{R}$ is the intersection of the matroid polytope for M with the matroid base polytope for the partition matroid defined by the S_j sets for $j \in D'$. This polytope is known to have integral extreme points. Similar to Step II, we define a linear function $H(z) = \sum_i f_i z_i + \sum_{k \in D} A_k(z)$, where

$$A_k(z) = \begin{cases} \sum_{i \in S_{\mathsf{ctr}(k)}} d'_k c_{ik} z_i & \text{if } i_1(k) \in S_{\mathsf{ctr}(k)} \\ \sum_{i \in S_{\mathsf{ctr}(k)}} d'_k c_{ik} z_i + d'_k \big(c_{i_1(k)k} - c_{i_2(k)k}\big) z_{i_1(k)} & \text{otherwise.} \end{cases}$$

Since $\mathcal{R}$ is integral, we can find an integer point $\tilde{y} \in \mathcal{R}$ such that $H(\tilde{y}) \leq H(\hat{y}')$. This yields an integer solution $(\tilde{x}, \tilde{y})$ to the instance with client set D, where we assign each client $j \in D'$ to the unique facility opened from S_j, and each client $k \in D \setminus D'$ either to $i_1(k)$ if it is open (i.e., $\tilde{y}_{i_1(k)} = 1$), or to the facility opened from $S_{\mathsf{ctr}(k)}$. In Lemma 3.3 we prove that the cost of this integer solution is at most $H(\tilde{y})$, and in Lemma 3.4 we show that $H(\hat{y}')$ is at most twice the cost of $(\hat{x}, \hat{y})$ and hence, at most $6 \cdot OPT$ (by Lemma 3.2). Combined with Lemma 3.1, this yields Theorem 3.5.

▶ **Lemma 3.3.** *The cost of* $(\tilde{x}, \tilde{y})$ *is at most* $H(\tilde{y}) \leq H(\hat{y}')$.

▶ **Lemma 3.4.** $H(\hat{y}')$ *is at most twice the cost of* $(\hat{x}, \hat{y})$.

▶ **Theorem 3.5.** *The integer solution* $(\tilde{x}, \tilde{y})$ *translates to an integer solution to the original instance of cost at most* $10 \cdot OPT$.

3.4 Improvement to 8-Approximation

The procedure described in Section 3.3 shows that *any* half-integral solution can be rounded to an integral one losing a factor of 2 in the cost. We obtain an improved approximation ratio of 8 by exploiting the structure leading to the half-integral solution obtained in Section 3.2. The key to the improvement comes from the following observation (in various flavors). Consider a non-cluster-center $k \in D' \setminus D$ with $\mathsf{ctr}(k) = j$. Let i be a facility serving both j and k.

Suppose i is not the primary facility of k. Without any further information, we can only say that $c_{jk} \leq c_{ij} + c_{ik} \leq 3\gamma_j + 3\gamma_k$. However, if we define our half-integral solution by setting the secondary facility of k to be the primary facility of the client (in D) nearest to k, then we have the better bound $c_{jk} \leq 2\gamma_j + 2\gamma_k$, which yields an improved bound for k's assignment cost. To push this observation through, we will "couple" the rounding steps used to obtain the half-integral and integral solutions: we tailor the function $T(.)$ (defined in Step II above) so as to allow one to bound the total cost of the final integral solution obtained. Also, we use a different criterion for selecting a cluster center in the clustering performed in Step III.

The first step is the same as Step I in Section 3.2. Recall that the new client-set is D with demands $\{d'_j\}_{j\in D}$, OPT' is the cost of (x, y) for the modified instance, and for each $j \in D$ we define $F_j = \{i : c_{ij} = \min_{k\in D} c_{ik}\}$, $F'_j = \{i \in F_j : c_{ij} \leq 2\bar{C}_j\}$, $\gamma_j = \min_{i\notin F_j} c_{ij}$, and $G_j = \{i \in F_j : c_{ij} \leq \gamma_j\}$.

A1. Obtaining a Half-integral Solution. Set $y'_i = x_{ij} \leq y_i$ if $i \in G_j$, and $y'_i = 0$ otherwise. We define $T(v) = \sum_i f_i v_i + \sum_j d'_j\big(2\sum_{i\in G_j} c_{ij}v_i + 4\gamma_j(1 - \sum_{i\in G_j} v_i)\big)$ for $v \in \mathbb{R}^{\mathcal{F}}_+$ with some hindsight. Since y' lies in the half-integral polytope $\mathcal{P}$ (see (1)), we can obtain a half-integral $\hat{y}$ such that $T(\hat{y}) \leq T(y')$.

For each client $j \in D$, define $\sigma(j) = j$ if $\hat{y}(G_j) = 1$, and $\sigma(j) = \arg\min_{k\in D:k\neq j} c_{jk}$ otherwise (breaking ties arbitrarily). Note that $c_{j\sigma(j)} \leq 2\gamma_j$. As before, we call the facility i nearest to j with $\hat{y}_i > 0$ the primary facility of j and denote it by $i_1(j)$; we set $\hat{x}_{i_1(j)j} = \hat{y}_{i_1(j)}$. Note that $i_1(j) \in F'_j$. If $\hat{y}_{i_1(j)} < 1$ and $\hat{y}(G_j) = 1$, let i' be the fractionally open facility other than $i_1(j)$ nearest to j; otherwise, if $\hat{y}_{i_1(j)} < 1$ and $\hat{y}(G_j) < 1$, (so $\sigma(j) \neq j$ and $\hat{y}_{i_1(j)} = \frac{1}{2}$), let i' be the primary facility of $\sigma(j)$. We call i' the secondary facility of j, and denote it by $i_2(j)$. Again, for convenience, we consider j as having both a primary and secondary facility and $\hat{x}_{i_1(j)j} = \hat{x}_{i_2(j)j} = \frac{1}{2}$, with the understanding that if $\hat{y}_{i_1(j)} = 1$, then $i_2(j) = i_1(j)$ and $\hat{x}_{i_1(j)j} = 1$. Let $S_j = \{i : \hat{x}_{ij} > 0\} = \{i_1(j), i_2(j)\}$.

A2. Clustering and Rounding to an Integral Solution. For each $j \in D$, define $C'_j = \big(c_{i_1(j)j} + c_{j\sigma(j)} + c_{i_2(j)\sigma(j)}\big)/2$. We cluster clients as in Step III in Section 3.3, except that we repeatedly pick the client with smallest C'_j among the remaining clients to be the cluster center. As before, let D' denote the set of cluster centers, and let $\mathsf{ctr}(k) = j \in D'$ for $k \in D$ if k was removed in the clustering process because j was chosen as a cluster center and $S_j \cap S_k \neq \emptyset$.

Similar to Step IV in Section 3.3, for each $i \in \mathcal{F}$, define $\hat{y}'_i = \hat{x}_{ij} \leq \hat{y}_i$ if $i \in S_j$ where $j \in D'$ and $\hat{y}'_i = \hat{y}_i$ otherwise. For $z \in \mathbb{R}^{\mathcal{F}}_+$, define $H(z) = \sum_i f_i z_i + \sum_{k\in D} L_k(z)$, where $L_k(z)$ is $\sum_{i\in S_{\mathsf{ctr}(k)}} d'_k c_{ik} z_i$ if $i_1(k) \in S_{\mathsf{ctr}(k)}$, and $\sum_{i\in S_{\mathsf{ctr}(k)}} d'_k\big(c_{k\sigma(k)} + c_{i\sigma(k)}\big)z_i + d'_k\big(c_{i_1(k)k} - c_{k\sigma(k)} - c_{i_1(\sigma(k))\sigma(k)}\big)z_{i_1(k)}$ otherwise. Since $\hat{y}'$ lies in the integral polytope $\mathcal{R}$ (see (2)), we can obtain an integral vector $\tilde{y}$ such that $H(\tilde{y}) \leq H(\hat{y}')$, and a corresponding integer solution $(\tilde{x}, \tilde{y})$ (as in Step IV in Section 3.3).

Analysis. The 8-approximation guarantee (Theorem 3.8) follows directly by combining Lemmas 3.6 and 3.7 with Lemma 3.1.

▶ **Lemma 3.6.** *We have $T(\hat{y}) \leq T(y') \leq 4 \cdot OPT' \leq 4 \cdot OPT$.*

Proof. We know that $T(\hat{y}) \leq T(y')$ and $OPT' \leq OPT$. We have $OPT' = \sum_i f_i y_i + \sum_j d'_j \bar{C}_j$, and for any $j \in D$, we have $\bar{C}_j = \sum_{i\in G_j} c_{ij}x_{ij} + \sum_{i\notin G_j} c_{ij}x_{ij} \geq \sum_{i\in G_j} c_{ij}x_{ij} + \gamma_j(1 - \sum_{i\in G_j} x_{ij})$ by the definition of γ_j. So $T(y')$ is at most

$$\sum_i f_i y_i + \sum_j d'_j\Big(\sum_{i\in G_j} c_{ij}x_{ij} + 4\gamma_j(1 - \sum_{i\in G_j} x_{ij})\Big) \leq \sum_i f_i y_i + 4\sum_j d'_j \bar{C}_j. \qquad \blacktriangleleft$$

▶ **Lemma 3.7.** *The cost of $(\tilde{x}, \tilde{y})$ is at most $H(\tilde{y}) \le H(\hat{y}')$, and $H(\hat{y}') \le T(\hat{y})$.*

Proof. We first argue that the cost of $(\tilde{x}, \tilde{y})$ is at most $H(\tilde{y})$. The facility opening cost is $\sum_i f_i \tilde{y}_i$. The assignment cost of a client $j \in D'$ is $\sum_{i \in S_j} d'_j c_{ij} \tilde{y}_i = L_j(\tilde{y})$. Consider a client $k \in D \setminus D'$ with $\mathsf{ctr}(k) = j$. Let $i' = i_1(j),\ i'' = i_2(j)$. If $\tilde{y}_{i_1(k)} = 0$ or $i_1(k) \in S_j$, then $L_k(\tilde{y})$ is at least $d'_k \sum_{i \in S_j} c_{ik} \tilde{y}_i$. So suppose $\tilde{y}_{i_1(k)} = 1$ and $i_1(k) \notin S_j$. Then the assignment cost of k is $d'_k c_{i_1(k)k}$, and since $c_{i\sigma(k)} \ge c_{i_1(\sigma(k))\sigma(k)}$ for every $i \in S_j$, we have $L_k(\tilde{y}) \ge d'_k c_{i_1(k)k}$.

We now show that $H(\hat{y}') \le T(\hat{y})$. Define $B_j(\hat{y}) := d'_j\big(2\sum_{i \in G_j} c_{ij}\hat{y}_i + 4\gamma_j(1 - \hat{y}(G_j))\big)$. So $T(\hat{y}) = \sum_i f_i \hat{y}_i + \sum_{j \in D} B_j(\hat{y})$. Clearly $\sum_i f_i \hat{y}'_i \le \sum_i f_i \hat{y}_i$. We show that $L_j(\hat{y}') \le B_j(\hat{y})$ for every $j \in D$, which will complete the proof.

We first argue that $d'_j C'_j \le B_j(\hat{y})$ for every $j \in D$. If $\hat{y}(G_j) = 1$, then $d'_j C'_j = \sum_{i \in G_j} d'_j c_{ij} \hat{y}_i \le B_j(\hat{y})$. Otherwise, $\hat{y}(G_j) = \frac{1}{2}$, and $c_{j\sigma(j)} + c_{i_1(\sigma(j))\sigma(j)} \le 3\gamma_j$; so $d'_j C'_j \le d'_j\big(\sum_{i \in G_j} c_{ij}\hat{y}_i + 3\gamma_j(1 - \hat{y}(G_j))\big) \le B_j(\hat{y})$.

For a client $j \in D'$, we have $L_j(\hat{y}') = d'_j\big(c_{i_1(j)j} + c_{i_2(j)j}\big)/2 \le d'_j C'_j \le B_j(\hat{y})$. Now consider a client $k \in D \setminus D'$. Let $j = \mathsf{ctr}(k)$, and $i' = i_1(j),\ i'' = i_2(j)$. Note that $C'_j \le C'_k$. We consider two cases.

1. $i_1(k) \in S_j$. This means that $i_1(k) = i'' \ne i'$ and $k = \sigma(j)$. So
$$L_k(\hat{y}') = \frac{d'_k}{2} \cdot \big(c_{i''k} + c_{i'k}\big) \le \frac{d'_k}{2} \cdot \big(c_{i'j} + c_{jk} + c_{i''k}\big) = d'_k C'_j \le d'_k C'_k \le B_k(\hat{y}).$$
2. $i_1(k) \notin S_j$. This implies that $\hat{y}(G_k) = \hat{y}_{i_1(k)} = \frac{1}{2}$. Let $\ell = \sigma(k)$ (which is the same as j if $i_2(k) = i_1(j)$). We have $L_k(\hat{y}') = \frac{d'_k}{2} \cdot \big(2c_{k\ell} + c_{i'\ell} + c_{i''\ell} + c_{i_1(k)k} - c_{k\ell} - c_{i_1(\ell)\ell}\big)$. If $\ell = j$, then $L_k(\hat{y}') = \frac{d'_k}{2} \cdot \big(c_{i_1(k)k} + c_{jk} + c_{i''j}\big)$. Notice that $c_{i''j} \le 2C'_j - c_{i'j}$. So we obtain that
$$L_k(\hat{y}') \le \frac{d'_k}{2} \cdot \big(c_{i_1(k)k} + c_{jk} + 2C'_j - c_{i'j}\big) \le \frac{d'_k}{2} \cdot \big(c_{i_1(k)k} + c_{jk} + 2C'_k - c_{i'j}\big) = d'_k\big(c_{i_1(k)k} + c_{jk}\big).$$
If $\ell \ne j$, then $i_2(j) = i'' = i_2(k) = i_1(\ell)$, so $\ell = \sigma(j)$, and $c_{i'_j} + c_{j\ell} + c_{i''\ell} = 2C'_j \le 2C'_k = c_{i_1(k)k} + c_{k\ell} + c_{i''\ell}$. So $L_k(\hat{y}') \le \frac{d'_k}{2} \cdot \big(c_{i_1(k)k} + c_{k\ell} + c_{j\ell} + c_{i'j}\big) \le d'_k(c_{i_1(k)k} + c_{k\ell})$. In both cases,
$$L_k(\hat{y}') \le d'_k(c_{i_1(k)k} + c_{k\sigma(k)}) \le d'_k\Big(2\sum_{i \in G_k} c_{ik}\hat{y}_i + 4\gamma_k\big(1 - \hat{y}(G_k)\big)\Big) = B_k(\hat{y}). \qquad \blacktriangleleft$$

▶ **Theorem 3.8.** *The integer solution $(\tilde{x}, \tilde{y})$ translates to an integer solution to the original instance of cost at most $8 \cdot OPT$.*

▶ Remark. It is easy to modify the above algorithm to obtain a so-called *Lagrangian-multiplier preserving* (LMP) 8-approximation algorithm, that is, where the solution $(\tilde{x}, \tilde{y})$ returned satisfies $8\sum_i f_i \tilde{y}_i + \sum_{j \in \mathcal{D}, i} d_j c_{ij} \tilde{x}_{ij} \le 8 \cdot OPT$. To obtain this, the only change is that we redefine
$$T(v) = 8\sum_i f_i v_i + \sum_j d'_j\big(2\sum_{i \in G_j} c_{ij} v_i + 4\gamma_j(1 - \sum_{i \in G_j} v_i)\big), \qquad H(z) = 8\sum_i f_i z_i + \sum_{k \in D} L_k(z).$$
We now have $T(\hat{y}) \le T(y') \le 8\sum_i f_i y_i + 4\sum_{j \in D} d'_j \bar{C}_j$, and $8\sum_i f_i \tilde{y}_i + \sum_{j \in D, i} d'_j c_{ij} \tilde{x}_{ij} \le H(\tilde{y}) \le H(\hat{y}')$. Also, as before, we have $H(\hat{y}') \le T(\hat{y})$. Thus, we have
$$\begin{aligned} 8\sum_i f_i \tilde{y}_i + \sum_{j \in \mathcal{D}, i} d_j c_{ij} \tilde{x}_{ij} &\le 8\sum_i f_i \tilde{y}_i + \sum_{j \in D, i} d'_j c_{ij} \tilde{x}_{ij} + \sum_{j \in \mathcal{D} \setminus D} 4 d_j \bar{C}_j \\ &\le 8\sum_i f_i y_i + 4\sum_{j \in D} d_j \bar{C}_j + 8\sum_{j \in \mathcal{D} \setminus D} d_j \bar{C}_j \le 8 \cdot OPT. \end{aligned}$$

4 Extensions

4.1 Matroid Median with Two Matroids

A natural extension of matroid median is the *matroid-intersection median* problem, wherein are given two matroids on the facility-set $\mathcal{F}$, and we require the set of open facilities to be an independent set in both matroids. This problem turns out to be inapproximable to within any multiplicative factor in polytime.

▶ **Theorem 4.1.** *It is NP-complete to decide if an instance of matroid-intersection median has a zero-cost solution; this holds even if one of the matroids is a partition matroid. Hence, no multiplicative approximation is achievable in polytime for this problem unless P=NP.*

Proof. The reduction is from the *NP*-complete *directed Hamiltonian path* problem, wherein we are given a directed graph $D = (N, A)$, and two nodes s, t, and we need to determine if there is a simple (directed) $s \rightsquigarrow t$ path spanning all the nodes. The facility-set in the matroid-intersection median problem is the arc-set A, and every node except t is a client. One of the matroids M is the graphic matroid on the undirected version of D, that is, an arc-set is independent if it is acyclic when we ignore the edge directions. The second matroid M_2 is a partition matroid that enforces that every node other than s has at most one incoming arc. All facility-costs are 0. We set $c_{ij} = 0$ if i is an outgoing arc of j, and ∞ otherwise. Notice that this forms a metric since the sets $\{i : c_{ij} = 0\}$ are disjoint for different clients.

It is easy to see that an $s \rightsquigarrow t$ Hamiltonian path translates to a zero-cost solution to the matroid-intersection median problem. Conversely, if we have a zero-cost solution to matroid-intersection median, then it must open $|N| - 1$ facilities, one for each client. Hence, the resulting edges must form a (spanning) arborescence rooted at s, and moreover, every node other than t must have an outgoing arc. Thus, the resulting edges yield an $s \rightsquigarrow t$ Hamiltonian path. ◀

We consider two extensions of matroid median that are essentially special cases of matroid-intersection median and can be used to model some interesting problems (see Section 5). The techniques developed in Section 3 readily extend and yield an 8-approximation algorithm (in fact, an LMP 8-approximation) for both problems. These extensions may be viewed in some sense as the most-general special cases of matroid-intersection median that one can hope to approximately solve in polytime.

The setup in both extensions is similar. We have a matroid $M = (\mathcal{F}, \mathcal{I})$ on the facility-set (and clients with demands and assignment costs). $\mathcal{F}$ is partitioned into $\mathcal{F}_1 \cup \mathcal{F}_2$ and clients may only be assigned to facilities in $\mathcal{F}_1$; this can be encoded by setting $c_{ij} = \infty$ for all $i \in \mathcal{F}_2$ and $j \in \mathcal{D}$. We also have lower and upper bounds $(lb1, ub1)$, $(lb2, ub2)$, and (lb, ub) on the number of facilities that may be opened from $\mathcal{F}_1$, $\mathcal{F}_2$, and $\mathcal{F}$ respectively. We need to open a feasible set of facilities and assign every client to an open facility so as to minimize the total facility-opening and client-assignment cost. A set $F \subseteq \mathcal{F}$ of facilities is said to be feasible if: (i) $F \in \mathcal{I}$; (ii) $lb1 \leq |F \cap \mathcal{F}_1| \leq ub1$, $lb2 \leq |F \cap \mathcal{F}_2| \leq ub2$, $lb \leq |F| \leq ub$; and (iii) $F \cap \mathcal{F}_2$ satisfies problem-specific constraints. While the role of $\mathcal{F}_2$ may seem unclear, notice that a non-trivial lower bound on the number of $\mathcal{F}_2$-facilities imposes restrictions on the facilities that may be opened from $\mathcal{F}_1$ due to the matroid M (see, e.g., k-median forest in Section 5).

Two-matroid Median (2MMed). In addition to the above setup, we have another matroid $M_2 = (\mathcal{F}_2, \mathcal{I}_2)$ on $\mathcal{F}_2$ with rank function r_2. A set F of facilities is feasible if it satisfies (i) and (ii) above, and (iii) $F \cap \mathcal{F}_2 \in \mathcal{I}_2$. We may modify the matroids M and M_2 to incorporate

the upper bounds *ub* and *ub2* respectively in their definition; *we assume that this has been done in the sequel*. The LP-relaxation for 2MMed is quite similar to (P). We augment (P) with the constraints:

$$y(S) \le r_2(S) \quad \forall S \subseteq \mathcal{F}_2, \qquad lb1 \le y(\mathcal{F}_1) \le ub1, \quad lb2 \le y(\mathcal{F}_2), \quad lb \le y(\mathcal{F}).$$

Let (x, y) denote an optimal solution to this LP, and OPT denote its cost. The rounding procedure dovetails the one in Section 3. The first step is again Step I in Section 3.2. Let D be the new client-set with demands $\{d'_j\}_{j\in D}$, OPT' be the new cost of (x, y), and for each $j \in D$, we define F_j, F'_j, γ_j, and G_j as before. Note that $F_j \subseteq \mathcal{F}_1$ for all $j \in D$. A slight technicality arises in mimicking Step A1 in Section 3.4: setting $y'_i = x_{ij}$ for some facility $i \in G_j$ need not satisfy the lower-bound constraints. We deal with this by "cloning" facilities suitably to obtain: (i) a new $\mathcal{F}'_1$-set $\mathcal{F}_1$, corresponding facility-set $\mathcal{F}' = \mathcal{F}'_1 \cup \mathcal{F}_2$ and facility-opening vector $y \in \mathbb{R}_+^{\mathcal{F}'}$; (ii) a new set $G'_j \subseteq \mathcal{F}'_1$ for all $j \in D$; (iii) a new rank function $r' : 2^{\mathcal{F}'} \mapsto \mathbb{Z}_+$.

We continue with steps A1, A2 in Section 3.4, replacing G_j with G'_j, and using suitable polytopes in place of $\mathcal{P}$ and $\mathcal{R}$ to obtain the half-integral and integral solutions. To obtain a half-integral solution, we define

$$\mathcal{P}' := \left\{ v \in \mathbb{R}_+^{\mathcal{F}'} : \begin{array}{l} v(S) \le r'(S) \ \ \forall S \subseteq \mathcal{F}', \quad v(S) \le r_2(S) \ \ \forall S \subseteq \mathcal{F}_2, \quad lb \le v(\mathcal{F}') \\ lb1 \le v(\mathcal{F}'_1) \le ub1, \quad lb2 \le v(\mathcal{F}_2), \quad v(F'_j) \ge \frac{1}{2}, \ v(G'_j) \le 1 \quad \forall j \in D \end{array} \right\} \tag{3}$$

which contains y. The key observation is that an extreme point of $\mathcal{P}'$ is again defined by a linearly independent system of tight constraints coming from two laminar systems: one consisting of some tight $v(S) \le r'(S)$ and $lb \le v(\mathcal{F}') \le ub$ constraints; the other consisting of some tight $v(S) \le r_2(S)$ and $lb1 \le v(\mathcal{F}'_1) \le ub1$, $lb2 \le v(\mathcal{F}_2) \le ub2$ constraints, and some tight $v(F'_j) \le \frac{1}{2}$ and $v(G'_j) \ge 1$ constraints. Thus, $\mathcal{P}'$ has half-integral extreme points, and so we can find a half-integral $\hat{y}$ such that $T(\hat{y}) \le T(y)$, and a corresponding solution $(\hat{x}, \hat{y})$. We round this to an integral solution as in step A2, using the polytope

$$\mathcal{R}' := \left\{ z \in \mathbb{R}_+^{\mathcal{F}'} : \begin{array}{l} z(S) \le r'(S) \quad \forall S \subseteq \mathcal{F}', \qquad z(S) \le r_2(S) \quad \forall S \subseteq \mathcal{F}_2 \\ lb1 \le z(\mathcal{F}'_1) \le ub1, \quad lb2 \le z(\mathcal{F}_2), \quad lb \le z(\mathcal{F}'), \quad z(S_j) = 1 \ \ \forall j \in D' \end{array} \right\} \tag{4}$$

which has integral extreme points. A useful observation is that if $j \in D'$ then we may assume that $\hat{x}_{ij} = \hat{y}_i$ for all $i \in S_j$, and so $\hat{y} \in \mathcal{R}'$. So we obtain an integral vector $\tilde{y}$ such that $H(\tilde{y}) \le H(\hat{y})$, and hence an integer solution $(\tilde{x}, \tilde{y})$. (Here $T(.)$ and $H(.)$ are as defined in in Section 3.4.) Mimicking Lemmas 3.6 and 3.7, we obtain that $T(\hat{y}) \le T(y) \le 4 \cdot OPT'$, and the cost of $(\tilde{x}, \tilde{y})$ is at most $H(\tilde{y}) \le H(\hat{y}) \le T(\hat{y})$. Thus, we obtain the following theorem.

▶ **Theorem 4.2.** *The integer solution $(\tilde{x}, \tilde{y})$ yields an integer solution to 2MMed of cost at most $8 \cdot OPT$.*

Laminarity-constrained Matroid Median (LCMMed). In LCMMed, in addition to the common setup, we have a laminar family $\mathcal{L}$ on $\mathcal{F}_2$ and bounds $0 \le \ell_S \le u_S$ for every set $S \in \mathcal{L}$; a set F of facilities is feasible if it satisfies (i) and (ii) above, and (iii) $\ell_S \le |F \cap S| \le u_S$ for all $S \in \mathcal{L}$, The approach used for 2MMed also works for LCMMed. The only (obvious) changes are that the LP-relaxation, as well as the definition of the polytopes $\mathcal{P}'$ and $\mathcal{R}'$ (in (3) and (4)) now include the laminarity constraints in place of the rank constraints for the second matroid. All other steps and arguments proceed *identically*, and so we obtain an 8-approximation algorithm for laminarity-constrained matroid median.

4.2 Matroid Median with Penalties

This is the generalization of matroid median where are allowed to leave some clients unassigned at the expense of incurring a penalty $d_j\pi_j$ for each unassigned client j. This changes the LP-relaxation (P) as follows. We use a variable z_j for each client $j \in \mathcal{D}$ to denote if we incur the penalty for client j, and modify the assignment constraint for client j to $\sum_i x_{ij} + z_j \geq 1$; also the objective is now to minimize $\sum_i f_i y_i + \sum_j d_j \left(\sum_i c_{ij} x_{ij} + \pi_j z_j\right)$. Let (x, y, z) denote an optimal solution to this LP and OPT be its value. Krishnaswamy et al. [11] showed that (x, y, z) can be rounded to an integer solution losing a factor of 360. We show that our rounding approach for matroid median can be adapted to yield a substantially improved 24-approximation algorithm. The rounding procedure is similar to the one described in Section 3 for matroid median, except that we now need to deal with the complication that a client need be assigned fractionally to an extent of 1. We defer the algorithm description and its analysis to the full version.

5 Applications

We now show that the various facility location problems listed below can be cast as special cases of matroid median or the extensions considered in Section 4.1. Thus, our 8-approximation algorithms for matroid median and these extensions immediately yield *improved approximation guarantees for all these problems.*

Problem	**Previous best approximation factor**
Data placement problem [2, 3]	10 [3]
Mobile facility location [9, 1] (with general movement costs)	—; *our reduction* and results of [11, 6] yield factors of 16 and 9 $((3+\epsilon)$ [1] for proportional movement costs)
k-median forest [10] (with non-uniform metrics)	16 [10] $((3+\epsilon)$ [10] for related metrics)
Metric-uniform minimum-latency UFL (MLUFL) [4]	10.773 [4]

The Data Placement Problem. We have a set of caches $\mathcal{F}$, a set of data objects $\mathcal{O}$, and a set of clients $\mathcal{D}$. Each cache $i \in \mathcal{F}$ has a capacity u_i. Each client $j \in \mathcal{D}$ has demand d_j for a specific data object $o(j) \in \mathcal{O}$ and has to be assigned to a cache that stores $o(j)$. Storing an object o in cache i incurs a storage cost of f_i^o, and assigning client j to cache i incurs an access cost of $d_j c_{ij}$, where the c_{ij}s form a metric. We want to determine a set of objects $\mathcal{O}(i) \subseteq \mathcal{O}$ to place in each cache $i \in \mathcal{F}$ satisfying $|\mathcal{O}(i)| \leq u_i$, and assign each client j to a cache $i(j)$ that stores object $o(j)$, (i.e., $o(j) \in \mathcal{O}(i(j))$) so as to minimize $\sum_{i\in\mathcal{F}} \sum_{o\in\mathcal{O}(i)} f_i^o + \sum_{j\in\mathcal{D}} d_j c_{i(j)j}$.

Reduction to matroid median. The facility-set in the matroid-median instance is $\mathcal{F} \times \mathcal{O}$. Facility (i, o) denotes that we store object o in cache i, and has cost f_i^o. The client set is $\mathcal{D}$. We set the distance $c_{(i,o)j}$ to be c_{ij} if $o(j) = o$ and ∞ otherwise, thus enforcing that each client j is only assigned to a facility containing object $o(j)$. The new distances form a metric if the c_{ij}s form a metric. The cache-capacity constraints are incorporated via the matroid where a set $S \subseteq \mathcal{F} \times \mathcal{O}$ is independent if $|\{(i', o) \in S : i' = i\}| \leq u_i$ for every $i \in \mathcal{F}$.

Mobile facility location. In the version with general movement costs, the input is a metric space $\left(V, \{c_{ij}\}\right)$. We have a set $\mathcal{D} \subseteq V$ of clients, with each client j having demand d_j, and a set $\mathcal{F} \subseteq V$ of initial facility locations. A solution moves each facility $i \in \mathcal{F}$ to a final location $s_i \in V$ incurring a movement cost of $w_{is_i} \geq 0$, and assigns each client j to the final location

s of some facility incurring an assignment cost of $d_j c_{sj}$. The goal is to minimize the sum of all the movement and assignment costs.

Reduction to matroid median. We define the facility-set in the matroid-median instance to be $\mathcal{F} \times V$. Facility (i, s_i) denotes that $i \in \mathcal{F}$ is moved to location $s_i \in V$, and has cost w_{is_i} (note that s_i could be i). The client-set is unchanged, and we set $c_{(i,s_i)j}$ to be $c_{s_i j}$ for every facility $(i, s_i) \in \mathcal{F} \times V$ and client $j \in \mathcal{D}$. These new distances form a metric: we have $c_{(i,s_i)j} \leq c_{(i,s_i)k} + c_{(i',s_{i'})k} + c_{(i',s_{i'})j}$ since $c_{s_i j} \leq c_{s_i k} + c_{s_{i'} k} + c_{s_{i'} j}$. The constraint that a facility in $\mathcal{F}$ can only be moved to one final location can be encoded by defining a matroid where a set $S \subseteq \mathcal{F} \times V$ is said to be independent if $|\{(i', s) \in S : i' = i\}| \leq 1$ for all $i \in \mathcal{F}$.

k-median Forest. In the non-uniform version, we have two metric spaces $\big(V, \{c_{uv}\}\big)$ and $\big(V, \{d_{uv}\}\big)$. The goal is to find $S \subseteq V$ with $|S| \leq k$ and assign every node $j \in V$ to $i(j) \in S$ so as to minimize $\sum_j c_{i(j)j} + d\big(\mathsf{MST}(V/S)\big)$, where $\mathsf{MST}(V/S)$ is a minimum spanning forest where each component contains a node of S.

Reduction to 2MMed (or LCMMed). We actually reduce a generalization, where there is an "opening cost" $f_i \geq 0$ incurred for including i in S; the resulting instance is also an LCMMed instance. We add a root r to V. The facility-set $\mathcal{F}$ is the edge-set of the complete graph on $V \cup \{r\}$. The client-set is $\mathcal{D} := V$. Selecting a facility (r, i) denotes that $i \in S$, and selecting a facility (u, v), where $u, v \neq r$, denotes that (u, v) is part of $\mathsf{MST}(V/S)$. We let $\mathcal{F}_1$ be the edges incident to r, and $\mathcal{F}_2$ be the remaining edges. The cost of a facility $(r, i) \in \mathcal{F}_1$ is f_i; the cost of a facility $(u, v) \in \mathcal{F}_2$ is d_{uv}. The client-facility distances are given by $c_{(r,i)j} = c_{ij}$ and $c_{ej} = \infty$ for every $e \in \mathcal{F}_2$. Note that these $\{c_{ej}\}$ distances form a metric. We let M be the graphic matroid of the complete graph on $V \cup \{r\}$. We impose a lower bound of $|V|$ on the number of facilities opened from $\mathcal{F}$, and an upper bound of k on the number of facilities opened from $\mathcal{F}_1$. The matroid M_2 on $\mathcal{F}_2$ is the vacuous one where every set is independent.

A feasible solution to the 2MMed instance corresponds to a spanning tree on $V \cup \{r\}$ where r has degree at most k. This yields a solution to k-median forest of no-greater cost, where the set S is the set of nodes adjacent to r in this edge-set. Conversely, it is easy to see that a solution S to the k-median forest instance yields a 2MMed solution of no-greater cost.

Metric Uniform MLUFL. We have a set $\mathcal{F}$ of facilities with opening costs $\{f_i\}_{i \in \mathcal{F}}$, and a set $\mathcal{D}$ of clients with assignment costs $\{c_{ij}\}_{j \in \mathcal{D}, i \in \mathcal{F}}$, where the c_{ij}s form a metric. Also, we have a monotone latency-cost function $\lambda : \mathbb{Z}_+ \mapsto \mathbb{R}_+$. The goal is to choose a set $F \subseteq \mathcal{F}$ of facilities to open, assign each open facility $i \in F$ a distinct time-index $t_i \in \{1, \ldots, |\mathcal{F}|\}$, and assign each client j to an open facility $i(j) \in F$ so as to minimize $\sum_{i \in F} f_i + \sum_{j \in \mathcal{D}} \big(c_{i(j)j} + \lambda(t_{i(j)})\big)$.

Reduction to Matroid Median. We define the facility-set to be $\mathcal{F} \times \{1, \ldots, |\mathcal{F}|\}$ and the matroid on this set to encode that a set S is independent if $|\{(i, t') \in S : t' = t\}| \leq 1$ for all $t \in \{1, \ldots, |\mathcal{F}|\}$. We set $f_{(i,t)} = f_i$ and $c_{(i,t),j} = c_{ij} + \lambda(t)$; note that these distances form a metric. It is easy to see that we can convert any matroid-median solution to one where we open at most one (i, t) facility for any given i without increasing the cost, and hence, the matroid-median instance correctly encodes metric uniform MLUFL.

6 Knapsack Median

We now consider the *knapsack median problem* [11, 12], wherein instead of a matroid on the facility-set, we have a knapsack constraint on the facility-set. Kumar [12] obtained the first constant-factor approximation algorithm for this problem, and [6] obtained an improved

34-approximation algorithm. We consider a somewhat more-general version of knapsack median, wherein each facility i has a facility-opening cost f_i and a *weight* w_i, and we have a knapsack constraint $\sum_{i\in F} w_i \leq B$ constraining the total weight of open facilities. We leverage the ideas from our improved rounding procedure for matroid median to obtain an improved 32-approximation algorithm for this (generalized) knapsack-median problem.

We may assume that we know the maximum facility-opening cost f^{opt} of a facility opened by an optimal solution, so in the sequel we assume that $f_i \leq f^{opt}$, $w_i \leq B$ for all facilities $i \in \mathcal{F}$. Krishnaswamy et al. [11] showed that the natural LP-relaxation for knapsack median has a bad integrality gap; this holds even after augmenting the natural LP with knapsack-cover inequalities. To circumvent this difficulty, Kumar [12] proposed the following lower bound, which we also use. Suppose that we have an estimate C^{opt} within a $(1+\epsilon)$-factor of the connection cost of an optimal solution (which we can obtain by enumerating all powers of $(1+\epsilon)$). Then, defining $U_j := \arg\max\{z : \sum_k d_k \max\{0, z - c_{jk}\} \leq C^{opt}\}$, Kumar argued that the constraint $x_{ij} = 0$ if $c_{ij} > U_j$ is valid for the knapsack median instance. We augment the natural LP-relaxation with these constraints to obtain the following LP (K-P).

$$\begin{aligned}
\min \quad & \sum_i f_i y_i + \sum_j \sum_i d_j c_{ij} x_{ij} && && \text{(K-P)}\\
\text{s.t.} \quad & \sum_i x_{ij} = 1 && \forall j\\
& x_{ij} \leq y_i && \forall i,j\\
& \sum_i w_i y_i \leq B\\
& x_{ij}, y_i \geq 0 && \forall i,j; \qquad x_{ij} = 0 \quad \text{if } c_{ij} > U_j.
\end{aligned}$$

Let (x,y) be an optimal solution to (K-P) and OPT be its value. Let $\bar{C}_j = \sum_i c_{ij} x_{ij}$. Note that if our estimate C^{opt} is correct, then OPT is at most the optimal value *opt* for the knapsack median instance. We show that (x,y) can be rounded to an integer solution of cost $f^{opt} + 4C^{opt} + 28 \cdot OPT$. Thus, if consider all possible choices for C^{opt} in powers of $(1+\epsilon)$ and pick the solution returned with least cost, we obtain a solution of cost at most $(32+\epsilon)$ times the optimum. The rounding procedure first obtains a nearly half-integral solution whose cost is within a constant-factor of the optimum, which then turns out to be easy to round to an integral solution. The resulting algorithm and analysis is simpler than that in [12, 6]. A detailed description of our algorithm is as follows.

K1. Consolidating Demands. We start by consolidating demands as in Step I in Section 3.2. We now work with the client set D and the demands $\{d'_j\}_{j\in D}$. For $j \in D$, we use $M_j \subseteq \mathcal{D}$ to denote the set of clients (including j) whose demands were moved to j. Note that the M_js partition $\mathcal{D}$. Let OPT' denote the cost of (x,y) for this modified instance. As before, for each $j \in D$ we define $F_j = \{i : c_{ij} = \min_{k\in D} c_{ik}\}$, $F'_j = \{i \in F_j : c_{ij} \leq 2\bar{C}_j\}$, $\gamma_j = \min_{i\notin F_j} c_{ij}$, and $G_j = \{i \in F_j : c_{ij} \leq \gamma_j\}$.

K2. Obtaining a Nearly Half-integral Solution. Set $y'_i = x_{ij} \leq y_i$ if $i \in G_j$, and $y'_i = 0$ otherwise. Let $\mathcal{F}' = \bigcup_{j\in D} G_j$. In the sequel, we will only consider facilities in $\mathcal{F}'$. Consider the following polytope:

$$\mathcal{K} := \Big\{ v \in \mathbb{R}_+^{\mathcal{F}'} : v(F'_j) \geq \tfrac{1}{2}, \quad v(G_j) \leq 1 \quad \forall j \in D, \qquad \sum_i w_i v_i \leq B \Big\}. \tag{5}$$

Define $K(v) = \sum_i 2 f_i v_i + \sum_j d'_j \big(2 \sum_{i\in G_j} c_{ij} v_i + 8\gamma_j (1 - v(G_j))\big)$ for $v \in \mathbb{R}_+^{\mathcal{F}'}$. Since $y' \in \mathcal{K}$, we can efficiently obtain an extreme point $\hat{y}$ of $\mathcal{K}$ such that $K(\hat{y}) \leq K(y')$, the

support of $\hat{y}$ is a subset of the support of y', and all constraints that are tight under y' remain tight under $\hat{y}$. Thus, if $i \in G_j$ and $\hat{y}_i > 0$, then $y'_i > 0$ and so $c_{ij} \leq U_j$. Also, if $\hat{y}(G_j) < 1$ then $y'(G_j) < 1$, and so $\gamma_j \leq U_j$. We show in Lemma 6.1 that there is *at most one* client, which we call the *special client* and denote by s, such that G_s contains a facility i with $\hat{y}_i \notin \{0, \frac{1}{2}, 1\}$.

As in Section 3.4, for each client $j \in D$, define $\sigma(j) = j$ if $\hat{y}(G_j) = 1$, and $\sigma(j) = \arg\min_{k \in D: k \neq j} c_{jk}$ otherwise (breaking ties arbitrarily). Note that $c_{j\sigma(j)} \leq 2\gamma_j$. We now define the primary and secondary facilities of each client $j \in D$, which we denote by $i_1(j)$ and $i_2(j)$ respectively. If j is not the special client s, then $i_1(j)$ is the facility i nearest to j with $\hat{y}_i > 0$; otherwise, $i_1(j) = \arg\min_{i \in F'_j: \hat{y}_i > 0} w_i$ (breaking ties arbitrarily). If $\hat{y}_{i_1(j)} = 1$, then we set $i_2(j) = i_1(j)$. If $\hat{y}(G_j) < 1$, we set $i_2(j) = i_1(\sigma(j))$. If $\hat{y}_{i_1(j)} < \hat{y}(G_j) = 1$, we set $i_2(j)$ to: the half-integral facility in G_j other than $i_1(j)$ that is nearest to j if $j \neq s$; and the facility with smallest weight among the facilities $i \in G_j$ with $\hat{y}_i > 0$ (which could be the same as $i_1(j)$) if $j = s$. Define $S_j = \{i_1(j), i_2(j)\}$.

To gain some intuition, observe that the facilities $i_1(j)$ and $i_2(j)$ naturally yield a half-integral solution, where these facilities are open to an extent of $\frac{1}{2}$ and j is assigned to them to an extent of $\frac{1}{2}$; as before, if $i_1(j) = i_2(j)$, then this means that $i_1(j)$ is open to an extent of 1 and j is assigned completely to $i_1(j)$. The choice of the primary and secondary facilities ensures that this solution is feasible. (We do not however modify $\hat{y}$ as indicated above.)

K3. Clustering and Rounding to an Integral Solution. This step is quite straightforward. We define C'_j for $j \in D$, and cluster clients in D exactly as in step A2 in Section 3.4, and we open the facility with smallest weight within each cluster. Finally, we assign each client to the nearest open facility. Let $(\tilde{x}, \tilde{y})$ denote the resulting solution. Recall that D' is the set of cluster centers, and for $k \in D$, $\mathsf{ctr}(k)$ denotes the client in D due to which k was removed in the clustering process (so $\mathsf{ctr}(j) = j$ for $j \in D'$).

Analysis. We call a facility i half-integral (with respect to the vector $\hat{y}$ obtained in step K2) if $\hat{y}_i \in \{0, \frac{1}{2}, 1\}$ and fractional otherwise.

▶ **Lemma 6.1.** *The extreme point $\hat{y}$ of $\mathcal{K}$ obtained in step K2 is such that there is at most one client, called the* special client *and denoted by s, such that G_s contains fractional facilities. Moreover, if $\frac{1}{2} < \hat{y}(G_s) < 1$, then there is one exactly one facility $i \in F'_s$ such that $\hat{y}_i > 0$.*

Proof. Since $\hat{y}$ is an extreme point, it is well known that the submatrix A' of the constraint matrix whose columns correspond to the non-zero $\hat{y}_i$s and rows correspond to the tight constraints under $\hat{y}$ has full column-rank. The rows and columns of A' may be accounted for as follows. Each client $j \in D$ contributes: (i) a non-empty disjoint set of columns corresponding to the positive $\hat{y}_i$s in G_j; and (ii) a possibly-empty disjoint set of at most two rows corresponding to the tight constraints $\hat{y}(F'_j) = \frac{1}{2}$ and $\hat{y}(G_j) = 1$. This accounts for all columns of A'. There is at most one remaining row of A', which corresponds to the tight constraint $\sum_i w_i \hat{y}_i = B$.

Let p_j and q_j denote respectively the number of columns and rows contributed by $j \in D$. First, note that $p_j \geq q_j$ for all $j \in D$. This is clearly true if $q_j \leq 1$; if $q_j = 2$, then $\hat{y}(F'_j) = \frac{1}{2}$, $\hat{y}(G_j) = 1$, so both F'_j and G_j must have at least one positive $\hat{y}_i$. Also, note that if $p_j = q_j$, then G_j contains only half-integral facilities. Since $\sum_j p_j \leq \sum_j q_j + 1$, there can be at most one client such that $p_j > q_j$; we let this be our special client s. Note that we must have $p_s = q_s + 1$.

If $\frac{1}{2} < \hat{y}(G_s) < 1$ then: (i) $q_s = 0$, so $p_s = 1$; or (ii) $q_s = 1$, so $p_s = 2$, and since $\hat{y}(F'_s) = \frac{1}{2} < \hat{y}(G_s)$, both F'_s and G_s contain exactly one positive $\hat{y}_i$. ◀

It is easy to adapt the proof of Lemma 3.6, and obtain that $K(\hat{y}) \leq K(y') \leq 8 \cdot OPT' \leq 8 \cdot OPT$. Next, we prove our main result: the integer solution $(\tilde{x}, \tilde{y})$ computed is feasible and its cost for the modified instance is at most $K(\hat{y}) + f^{opt} + 4C^{opt} + 16 \cdot OPT$. Thus, "moving" the consolidated demands back to their original locations yields a solution of cost at most $(32 + \epsilon) \cdot opt$ for the correct guess of f^{opt} and C^{opt}. The following claims will be useful.

▶ **Claim 6.2.** *If $\hat{y}(G_j) = 1$ for some $j \in D$, then (we may assume that) j is a cluster center.*

Proof. Let $i' = i_1(j)$, $i'' = i_2(j)$. Let $k \in D$ be such that $S_k \cap S_j \neq \emptyset$. Then $\sigma(k) = j$. So $2(C'_k - C'_j) = c_{i_1(k)k} + c_{jk} - c_{i_2(j)j} \geq c_{i_1(k)j} - c_{i_2(j)j} \geq 0$ since $i_2(k) \notin G_j$. ◀

▶ **Claim 6.3.** *For any client $j \in D$, we have $d'_j U_j \leq C^{opt} + 4 \cdot OPT$.*

Proof. By definition, $\sum_k d_k \max\{0, U_j - c_{jk}\} \leq C^{opt}$. So $d'_j U_j = \sum_{k \in M_j} d_k U_j$, which equals

$$\sum_{k \in M_j} d_k (U_j - c_{jk}) + \sum_{k \in M_j} d_k c_{jk} \leq C^{opt} + \sum_{k \in M_j} 4 d_k \bar{C}_k \leq C^{opt} + 4 \cdot OPT. \qquad \blacktriangleleft$$

▶ **Theorem 6.4.** *The solution $(\tilde{x}, \tilde{y})$ computed in step K3 for the modified instance is feasible and has cost at most $K(\hat{y}) + f^{opt} + 4C^{opt} + 16 \cdot OPT$.*

Proof. Let $B_j(v) = d'_j \big(2 \sum_{i \in G_j} c_{ij} v_i + 8\gamma_j (1 - v(G_j))$ for $v \in \mathbb{R}_+^{\mathcal{F}'}$. So $K(\hat{y}) = 2 \sum_i f_i \hat{y}_i + \sum_j B_j(\hat{y})$. Recall that $S_j = \{i_1(j), i_2(j)\}$ for every $j \in D$.

We first prove feasibility and bound the total facility-opening cost. Consider a cluster centered at j. Let $i' = i_1(j)$, $i'' = i_2(j)$. Let $\hat{i}$ be the facility opened from S_j. If $\hat{y}(S_j) = 1$, then $w_{\hat{i}} \leq \sum_{i \in S_j} w_i \hat{y}_i$. Otherwise, either $j = s$ or $\sigma(j) = s$. If $j = \sigma(j) = s$, then $\hat{i}$ is the least-weight facility in G_j. Otherwise, if $j = s$ then $\hat{i}$ is the least-weight facility in $F'_j \cup \{i_2(j)\}$ and $\hat{y}(F'_j) + \hat{y}_{i_2(j)} \geq 1$; finally, if $j \neq \sigma(j) = s$ then $\hat{i}$ is the least-weight facility in $\{i_1(j)\} \cup F'_{\sigma(j)}$ and $\hat{y}_{i_1(j)} + \hat{y}(F'_{\sigma(j)}) \geq 1$. Since $S_j \subseteq G_j \cup G_{\sigma(j)}$, in every case, we have $w_{\hat{i}} \leq \sum_{i \in G_j \cup G_{\sigma(j)}} w_i \hat{y}_i$.

If all facilities in S_j are half-integral, then $f_{\hat{i}} \leq 2 \sum_{i \in S_j} f_i \hat{y}_i \leq 2 \sum_{i \in G_j \cup G_{\sigma(j)}} f_i \hat{y}_i$. Otherwise, we have $j = s$ or $\sigma(j) = s$, and we bound $f_{\hat{i}}$ by f^{opt}.

Note that if $k \in D'$ is some other cluster center, then $G_j \cup G_{\sigma(j)}$ is disjoint from $G_k \cup G_{\sigma(k)}$. If not, then we must have $\sigma(j) = k$ or $\sigma(k) = j$ or $\sigma(j) = \sigma(k)$, which yields the contradiction that $S_j \cap S_k \neq \emptyset$. So summing over all clusters, we obtain that the total weight of open facilities is at most $\sum_{j \in D'} \sum_{i \in G_j \cup G_{\sigma(j)}} w_i \hat{y}_i \leq \sum_i w_i \hat{y}_i \leq B$, and the facility opening cost is at most $2 \sum_i f_i \hat{y}_i + f^{opt}$.

We now bound the total client-assignment cost. Fix a client $j \in D'$. The assignment cost of j is at most $d'_j c_{i_2(j)j}$. Note that $c_{i_2(j)j} \leq 3U_j$. If $j \neq s$, then $B_j(\hat{y}) \geq d'_j c_{i_2(j)j}$: this holds if $\hat{y}(G_j) = 1$ since $\hat{y}_{i_2(j)} \geq \frac{1}{2}$; otherwise, $B_j(\hat{y}) \geq 4 d'_j \gamma_j \geq d'_j c_{i_2(j)j}$. If $j = s$, then its assignment cost is at most $3 d'_j U_j \leq 3C^{opt} + 12 \cdot OPT$ (Claim 6.3).

Now consider $k \in D \setminus D'$. Let $j = \mathsf{ctr}(k)$, and $i' = i_1(j)$, $i'' = i_2(j)$. We consider two cases.

1. $i_1(k) \in S_j$. Then $k = \sigma(j)$ and k's assignment cost is at most $d'_k c_{i_2(k)k}$. As above, this is bounded by $B_k(\hat{y})$ if $k \neq s$, and by $3C^{opt} + 12 \cdot OPT$ otherwise.
2. $i_1(k) \notin S_j$. Let $\ell = \sigma(k)$. We claim that the assignment cost of k is at most $d'_k \big(c_{i_1(k)k} + 4\gamma_k\big)$. To see this, first suppose $\ell \neq j$, and so $\ell = \sigma(j)$. Then, k's assignment cost is at most $d'_k \big(c_{k\ell} + c_{\ell j} + c_{i'j}\big) \leq d'_k \big(2c_{k\ell} + c_{i_1(k)k}\big) \leq d'_k \big(c_{i_1(k)k} + 4\gamma_k\big)$, where the first inequality follows since $C'_j \leq C'_k$. If $\ell = j$, then $i_2(k) = i_1(j) = i'$ and k's assignment cost is at most $d'_k \big(c_{jk} + c_{j\sigma(j)} + c_{i_2(j)\sigma(j)}\big) \leq d'_k \big(c_{i_1(k)} + 2c_{jk}\big) \leq d'_k \big(c_{i_1(k)k} + 4\gamma_k\big)$, where the first inequality again follows from $C'_j \leq C'_k$.

Since $k \notin D'$, we have $\hat{y}(G_k) < 1$ (by Claim 6.2). So $y'(G_k) < 1$ and $\gamma_k \leq U_k$. If $k \neq s$, then $B_k(\hat{y}) \geq d'_k\big(c_{i_1(k)k} + 4\gamma_k\big)$. If $k = s$ and $\hat{y}(G_k) = \frac{1}{2}$, then $B_k(\hat{y}) \geq 4d'_k\gamma_k$ and $d'_k c_{i_1(k)k} \leq d'_k U_k$. Otherwise, by Lemma 6.1, we have $\hat{y}_{i_1(k)} > \frac{1}{2}$, and so $B_k(\hat{y}) \geq d'_k c_{i_1(k)k}$ and $4d'_k\gamma_k \leq 4d'_k U_k$. Taking all cases into account, we can bound k's assignment cost by $B_k(\hat{y})$ if $k \neq s$, and by $B_k(\hat{y}) + 4d'_k U_k \leq B_k(\hat{y}) + 4C^{opt} + 16 \cdot OPT$ if $k = s$.

Putting everything together, the total cost of $(\tilde{x}, \tilde{y})$ is at most $2\sum_i f_i\hat{y}_i + \sum_j B_j(\hat{y}) + f^{opt} + 4C^{opt} + 16 \cdot OPT = K(\hat{y}) + f^{opt} + 4C^{opt} + 16 \cdot OPT$. ◀

▶ **Corollary 6.5.** *There is a $(32 + \epsilon)$-approximation algorithm for the knapsack median problem.*

Acknowledgments I thank Deeparnab Chakrabarty for various stimulating discussions that eventually led to this work. I thank Chandra Chekuri for some useful discussions regarding the matroid-intersection median problem.

References

1 S. Ahmadian, Z. Friggstad, and C. Swamy. Local-search based approximation algorithms for mobile facility location problems. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1607–1621, 2013.

2 I. Baev and R. Rajaraman. Approximation algorithms for data placement in arbitrary networks. In *Proceedings of the 12th SODA*, pages 661–670, 2001.

3 I. Baev, R. Rajaraman, and C. Swamy. Approximation algorithms for data placement problems. *SIAM Journal on Computing*, 38(4):1411–1429, 2008.

4 D. Chakrabarty and C. Swamy. Facility location with client latencies: linear-programming based techniques for minimum latency problems. In *Proceedings of the 15th IPCO*, pages 92–103, 2011.

5 M. Charikar, S. Guha, É. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k-median problem. *JCSS*, 65(1):129–149, 2002.

6 M. Charikar and S. Li. A dependent LP-rounding approach for the k-median problem. In *Proceedings of the 39th ICALP*, pages 194–205, 2012.

7 F. Chudak and D. Shmoys. Improved approximation algorithms for the uncapacitated facility location problem. *SIAM Journal on Computing*, 33(1):1–25, 2003.

8 W. Cook, W. Cunningham, W. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. John Wiley and Sons, Inc., New York, 1998.

9 Z. Friggstad and M. Salavatipour. Minimizing movement in mobile facility location problems. *ACM Transactions on Algorithms*, 7(3), 2011.

10 I. Gørtz and V. Nagarajan. Locating depots for capacitated vehicle routing. In *Proceedings of the 14th APPROX*, pages 230–241, 2011.

11 R. Krishnaswamy, A. Kumar, V. Nagarajan, Y. Sabharwal, and B. Saha. The matroid median problem. In *Proceedings, 22nd SODA*, pages 1117–1130, 2011.

12 A. Kumar. Constant-factor approximation algorithm for the knapsack median problem. In *Proceedings of the 23rd SODA*, pages 824–832, 2012.

13 D. B. Shmoys, É. Tardos, and K. I. Aardal. Approximation algorithms for facility location problems. In *Proceedings of the 29th STOC*, pages 265–274, 1997.

14 C. Swamy and D. B. Shmoys. Fault-tolerant facility location. In *Proceedings of the 14th SODA*, pages 735–736, 2003.

Robust Approximation of Temporal CSP*

Suguru Tamaki[1] and Yuichi Yoshida[2]

1 **Kyoto University**
Yoshida Honmachi, Sakyo-ku, Kyoto 606-8501, Japan
tamak@kuis.kyoto-u.ac.jp

2 **National Institute of Informatics and Preferred Infrastructure, Inc.**
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
yyoshida@nii.ac.jp

Abstract

A temporal constraint language Γ is a set of relations with first-order definitions in $(\mathbb{Q}; <)$. Let $\mathsf{CSP}(\Gamma)$ denote the set of constraint satisfaction problem instances with relations from Γ. $\mathsf{CSP}(\Gamma)$ admits robust approximation if, for any $\varepsilon \geq 0$, given a $(1-\varepsilon)$-satisfiable instance of $\mathsf{CSP}(\Gamma)$, we can compute an assignment that satisfies at least a $(1-f(\varepsilon))$-fraction of constraints in polynomial time. Here, $f(\varepsilon)$ is some function satisfying $f(0) = 0$ and $\lim_{\varepsilon \to 0} f(\varepsilon) = 0$.

Firstly, we give a qualitative characterization of robust approximability: Assuming the Unique Games Conjecture, we give a necessary and sufficient condition on Γ under which $\mathsf{CSP}(\Gamma)$ admits robust approximation. Secondly, we give a quantitative characterization of robust approximability: Assuming the Unique Games Conjecture, we precisely characterize how $f(\varepsilon)$ depends on ε for each Γ. We show that our robust approximation algorithms can be run in almost linear time.

1998 ACM Subject Classification F.2.0 [Analysis of Algorithms and Problem Complexity]: General

Keywords and phrases constraint satisfaction, maximum satisfiability, approximation algorithm, hardness of approximation, infinite domain

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.419

1 Introduction

In the Constraint Satisfaction Problem (CSP), we are given a set of constraints over a set of variables, and the task is to decide whether there exists an assignment of values to the variables that satisfies all the constraints. CSP can express general combinatorial and temporal problems in artificial intelligence, computer science, discrete mathematics, operations research, and elsewhere [11, 23].

In this paper, we consider the Temporal CSP (TCSP), a particular class of CSP where variables represent times and constraints represent sets of allowed temporal relations among them. Formally, a *temporal relation* is a relation with a first-order definition in $(\mathbb{Q}; <)$. TCSP forms a fundamental and important class of CSP over infinite domains [4]. Since TCSP is NP-hard in general, one of the major line of research is to identify tractable subclasses and develop efficient algorithms for them. One of the standard way to define subclasses of TCSP is restricting constraint languages.

A *temporal constraint language*, denoted by Γ, is a finite set of temporal relations. $\mathsf{CSP}(\Gamma)$ denotes the set of TCSP instances where each instance consists of constraints from Γ.

* This work was supported in part by MEXT KAKENHI (24106003); JSPS KAKENHI (25240002, 26330011, 26730009); JST ERATO Kawarabayashi Large Graph Project.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 419–432

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

Polynomial-time algorithms have been developed for larger and larger classes of constraint languages, see, e. g., [27, 26, 21], whereas TCSP for several specific constraint languages are known to be NP-complete [13]. Building on previous works, Bodirsky and Kára [7] finally showed the complete complexity classification of TCSP. Namely, they obtain a necessary and sufficient condition on Γ under which $\mathsf{CSP}(\Gamma)$ is tractable. The proof technique relies on a machinery from universal algebra, which plays an important role when we investigate the computational complexity of CSP in various settings.

In this paper, we study the complexity of Max-TCSP, instead of satisfiability of TCSP. We are interested in *robust approximability* of TCSP. An algorithm is called a (c, s)-*approximation algorithm* for $\mathsf{CSP}(\Gamma)$ if, given any c-satisfiable instance (some assignment satisfies at least a c-fraction of constraints) of $\mathsf{CSP}(\Gamma)$, it outputs an assignment that satisfies at least an s-fraction of constraints. An algorithm is called a *robust approximation algorithm* for $\mathsf{CSP}(\Gamma)$ if it is $(1-\varepsilon, 1-f(\varepsilon))$-approximation algorithm for any $\varepsilon \geq 0$, where f is some function satisfying $f(0) = 0$ and $\lim_{\varepsilon \to 0} f(\varepsilon) = 0$. When we want to specify $f(\varepsilon)$, we call it a $f(\varepsilon)$-*robust approximation algorithm*. A robust approximation algorithm is *polynomial-time* if for any fixed $\varepsilon \geq 0$, it runs in polynomial-time. Note that if $\mathsf{CSP}(\Gamma)$ admits polynomial-time robust approximation, then satisfiability of $\mathsf{CSP}(\Gamma)$ is solvable in polynomial-time. However, the reverse statement does not hold in general. For example, $\mathsf{CSP}(\{<\})$ (also known as the Acyclic Graph Problem) is solvable in polynomial-time, but $(1-\varepsilon, 1/2+\varepsilon)$-approximation is known to be *UG-Hard* [14], i. e., NP-Hard under Khot's Unique Games Conjecture (UGC) [17].

The notion of robust approximation is natural and useful, e. g., let us consider the Correlation Clustering Problem [1], which is equivalent to $\mathsf{CSP}(\{=, \neq\})$. Here, a variable stands for a datum and a constraint $u = v$ (resp., $u \neq v$) means u and v is similar (resp., dissimilar). The objective is to find a partition of the data into groups that agrees as much as possible with the constraints. If we are given a data set with a perfect (satisfiable) partition, then we can find it easily. However, if a small fraction of constraints are wrongly given by some reason, e. g., measurement error, then recovering the optimal partition may become much harder. Motivated by such practical applications, it is natural to ask what class of constraint languages admits robust approximation.

Our Contribution

In this paper, we give a complete complexity classification of robust approximability of TCSP.

We say that a constraint language Γ is *trivial* if every instance of $\mathsf{CSP}(\Gamma)$ is satisfiable unless it contains an individual constraint that is unsatisfiable such as $x_i \neq x_i$. Informally, Γ is a *Horn equality constraint language* if each relation in Γ can be defined as a Horn formula whose atoms are of the form $x = y$. See Preliminaries for the more detailed definition.

We have the following qualitative characterization:

▶ **Theorem 1.** *Let Γ be a temporal constraint language. Then, $\mathsf{CSP}(\Gamma)$ admits polynomial-time robust approximation if either Γ is trivial or a Horn equality constraint language. Otherwise, it is UG-Hard to robustly approximate $\mathsf{CSP}(\Gamma)$.*

We also show a more fine-grained classification that almost tightly (up to logarithmic factor) characterizes how $f(\varepsilon)$ depends on ε.

Informally, Γ is a *negative equality constraint language* if each relation in Γ can be defined as a disjunction of negative literals or a single positive literal whose atoms are of the form $x = y$. See Preliminaries for the more detailed definition.

We have the following quantitative characterization:

▶ **Theorem 2.** *Let Γ be a Horn equality constraint language.*

1. *If Γ is not trivial, it is UG-Hard to compute $o(\sqrt{\varepsilon})$-robust approximation of CSP(Γ).*
2. *If Γ is negative, there is a polynomial-time $O(\sqrt{\varepsilon}\log(1/\varepsilon))$-robust approximation algorithm for CSP(Γ).*
3. *If Γ is not negative, it is UG-Hard to compute $o(1/\log(1/\varepsilon))$-robust approximation of CSP(Γ).*
4. *There is a polynomial-time $O(\log\log(1/\varepsilon)/\log(1/\varepsilon))$-robust approximation algorithm for CSP(Γ).*

Here $O(\cdot)$ notation hides a constant depending on Γ.

Furthermore, we give almost linear time algorithms for the above mentioned robust approximation results.

▶ **Theorem 3.** *There exist algorithms that achieve the approximation guarantee mentioned in Items 2 and 4 of Theorem 2 in $O(m \cdot \mathrm{poly}\log n \cdot \exp(1/\varepsilon))$ time, where n is the number of variables and m is the number of constraints.*

Related Works

Motivated by obvious applications, CSP over *finite* domains has been a central problem in a lot of research areas. In their seminal paper [12], Feder and Vardi posed a famous dichotomy conjecture; "for any constraint language Γ over a finite domain, CSP(Γ) is either in P or NP-complete." The conjecture has been a driving force of the theoretical study of CSP and although it still remains open, we have developed deep mathematical insights on the structure of CSP, see, e. g., [9].

A systematic study of robust approximation algorithms was initiated by Zwick [28]. He gave polynomial-time robust approximation algorithms for 2SAT and Horn-SAT, which, combined with previous works [16, 24], implies a complete complexity classification of robust approximability of Boolean CSP. Later Dalmau and Krokhin [10] gave a more fine-grained classification which determines how $f(\varepsilon)$ depends on ε for each constraint language.

For CSP over general finite domains, Guruswami and Zhou conjectured that CSP(Γ) admits polynomial-time robust approximation if and only if CSP(Γ) has "bounded-width," which informally means that CSP(Γ) is solvable by a local consistency method. Dalmau and Krokhin [10], and Kun et al. [20] obtained robust approximation algorithms for the special case of width-1 and finally Barto and Kozik [2] confirmed the conjecture. Unlike Boolean CSP, a quantitative version of the classification has not been obtained so far, see [10].

As far as the authors know, there is only one paper that systematically studies the robust approximablity of CSP over infinite domains. Ordering CSP (OCSP) is TCSP with additional hard constraints that the variables need to be given different values. Guruswami et al. [14] showed that for any constraint language Γ, the best approximation algorithm for CSP(Γ) as OCSP is random assignment algorithms, assuming UGC. In particular, this implies that if Γ is nontrivial, then it is UG-hard to robustly approximate CSP(Γ) as OCSP. We notice that our results do not follow easily from [14] since the existence of hard constrains in OCSP affects the approximability of CSP.

As for specific CSP over infinite domains, we are only aware of the result for CSP($\{=, \neq\}$); Charikar et al. [8] gave a polynomial-time $O(\sqrt{\varepsilon}\log(1/\varepsilon))$-robust algorithm for it.

Our Technique

First we would like to emphasize that our contribution is the results themselves and not the techniques to prove them. Each technical proof is non-trivial but not too difficult to come

up with for experts on each topic such as universal algebra, approximation algorithms based on SDP, and connection between hardness of approximation and integrality gap. We briefly describe the overall proof structure below.

To prove Theorem 1, first we must identify the borderline which separates tractable and intractable cases. By the results of Bodirsky and Kára [6, 7] and Guruswami et al. [14], we see that if $\mathsf{CSP}(\Gamma)$ admits robust approximation, then Γ must be a Horn equality constraint language. Then, we show that Γ is a Horn equality constraint language is sufficient by giving robust approximation algorithms.

To prove Theorem 2, first we show that the "easiest" non-trivial TCSP is $\mathsf{CSP}(\{=,\neq\})$. The approximation hardness of $\mathsf{CSP}(\{=,\neq\})$ follows a simple reduction from Max-CUT. Next we extend the robust approximation algorithm for $\mathsf{CSP}(\{=,\neq\})$ due to Charikar et al. [8] and obtain an algorithm with the same approximation guarantee when Γ is negative. If Γ is not negative, we can show $\mathsf{CSP}(\Gamma)$ is as hard as $\mathsf{CSP}(\{\mathrm{ODD}_3,\neq\})$. The approximation hardness of $\mathsf{CSP}(\{\mathrm{ODD}_3,\neq\})$ follows by modifying the approximation hardness of Horn SAT due to Guruswami and Zhou [15].

Our algorithms are based on semidefinite programming (SDP) relaxation. One might think Raghavendra's canonical SDP relaxation for CSP over finite domains [22] can be extended to handle TCSP. This is the case in the sense that its integrality gap turns out to match UG-Hardness [14]. However, it is hard to explicitly analyze its approximation guarantee, and existing rounding techniques introduce errors depending on the domain size, which is huge for TCSP. Thus, we use an SDP relaxation tailored to equality constraint languages so as not to be affected by the domain size.

Our inapproximability results rely on UGC, which states that for any $\varepsilon > 0$, there exists an integer $q > 0$ such that it is NP-hard to compute $(1-\varepsilon,\varepsilon)$-approximation of CSP where each constraint is a two-variable linear equation over $\mathbb{Z}_q$. This complexity theoretic assumption enables us to prove *optimal* inapproximability results for various optimization problems such as Max-CUT, Vertex Cover etc., though proving them under P≠NP seems beyond our current proof techniques. See, e. g., [18] for discussion on UGC. To show inapproximability results in Theorem 2, we use the fact that the integrality gap matches UG-Hardness and explicitly give bad integrality gap instances.

Organization

In the next section, we introduce notion and standard tools to analyze TCSP. Then, we prove Theorem 1, which is a "qualitative" characterization of robust approximability. Next, we prove Theorem 2, which is a "quantitative" characterization of robust approximability. Finally, we prove Theorem 3, which gives almost linear time algorithms for the robust approximability results in Theorem 2.

2 Preliminaries

For an integer n, $[n]$ denotes the set $\{1,\dots,n\}$. We often use n and m to denote the number of variables and constraints of the instance we are concerned with, respectively.

For two real vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, $\angle(\boldsymbol{x},\boldsymbol{y})$ denotes the angle between them, i. e., $\arccos(\langle\boldsymbol{x},\boldsymbol{y}\rangle/(\|\boldsymbol{x}\|\cdot\|\boldsymbol{y}\|))$.

Temporal Constraint Language

A *temporal constraint language* Γ is a finite relational structure $(\mathbb{Q}; R_1, R_2, \dots)$ with a first-order definition in $(\mathbb{Q};<)$, the rational numbers with the dense linear order. Each R_i is a

temporal relation, i. e., $R_i \subseteq \mathbb{Q}^{k_i}$ for some finite k_i such that there is a first-order formula ϕ_i with k_i free variables that defines R_i over $(\mathbb{Q}; <)$.

An *instance* of the problem $\mathsf{CSP}(\Gamma)$ is $\mathcal{I} = (V, \mathcal{C})$, where V is a set of variables and $\mathcal{C}$ is a set of constraints. Each constraint $C \in \mathcal{C}$ is of the form $(x_1, \ldots, x_k; R)$, where $x_1, \ldots, x_k \in V$ are variables and $R \in \Gamma$ is a k-ary relation. We say that $\beta : V \to \mathbb{Q}$ *satisfies* a constraint $(x_1, \ldots, x_k; R) \in \mathcal{C}$ if the tuple $(\beta(x_1), \ldots, \beta(x_k))$ is in R. We say that β *satisfies* $\mathcal{I}$ if it satisfies all the constraints. When β satisfies a constraint C (resp., instance $\mathcal{I}$), we write $\beta \models C$ (resp., $\beta \models \mathcal{I}$). We denote by $\mathbf{opt}(\mathcal{I})$ the maximum fraction of constraints of $\mathcal{I}$ simultaneously satisfiable by some assignment.

An *equality constraint language* Γ is a temporal constraint language such that each relation can be defined with a $=$-formula, i. e., an AND-OR formula of atoms of the form $x = y$ or their negations.

For each relation R from an equality constraint language, we can find a formula ϕ_R of the equality relation that defines R. In particular, we can assume that ϕ_R is represented in conjunctive normal form. We say that R is *Horn* if each clause in ϕ_R contains at most one positive literal. We say that R is *negative* if each clause in ϕ_R consists of a single positive literal or a disjunction of negative literals. We say that an equality constraint language Γ is *Horn* (resp., negative) if every relation in Γ is Horn (resp., negative). The problem $\mathsf{CSP}(\Gamma)$ is called Horn $=$-SAT (resp., Negative $=$-SAT) if Γ is a Horn (resp., negative) equality constraint language Γ.

Universal Algebra

We introduce several definitions from universal algebra, which is a standard tool to investigate computational complexity of CSP.

An l-ary operation f *preserves* (or is a *polymorphism* of) a k-ary relation R if for any tuples $(a_1^i, \ldots, a_k^i) \in R$ $(i \in [l])$, the tuple $(f(a_1^1, \ldots, a_1^l), \ldots, f(a_k^1, \ldots, a_k^l))$ belongs to R as well. We say that f *preserves* (or is a *polymorphism* of) a constraint language Γ if f preserves all relations in Γ.

Let Γ be a constraint language and R be a relation. Then, R is *pp-definable* in Γ if R can be defined as $R(x_1, \ldots, x_k) = \exists y_1, \ldots, y_l(\psi(x_1, \ldots, x_k, y_1, \ldots, y_l))$, where ψ is a conjunction of atomic formulas with relations in Γ and the equality $=$. If ψ does not contain the equality $=$ then we say that R is *pp-definable in* Γ *without equality*. It is known that the set of relations pp-definable in Γ is exactly the set of relations whose polymorphisms are the same as Γ [3].

We introduce the notation $\mathsf{CSP}(\Gamma) \leq_{\mathrm{RA}} \mathsf{CSP}(\Gamma')$ as a shorthand for the following. For any error function f with $\lim_{\varepsilon \to 0} f(\varepsilon) = 0$ and $f(0) = 0$, if some polynomial-time algorithm $f(\varepsilon)$-robustly approximates $\mathsf{CSP}(\Gamma')$, then there is a polynomial-time algorithm that $O(f(\varepsilon))$-robustly approximates $\mathsf{CSP}(\Gamma)$.

Though the following lemma is originally proved for Boolean CSP, the proof is also valid for TCSP.

▶ **Lemma 4** ([10]). *Let Γ be a constraint language and let R be a relation pp-definable in Γ without equality. Then $\mathsf{CSP}(\Gamma \cup \{R\}) \leq_{\mathrm{RA}} \mathsf{CSP}(\Gamma)$.*

Thus, if Γ itself contains the equality relation, robust approximability of $\mathsf{CSP}(\Gamma)$ is determined by polymorphisms. Indeed, any non-trivial equality constraint language turns out to contain the equality relation. To show this, we use the following fact.

▶ **Lemma 5** ([6]). *Let Γ be an equality constraint language that is not preserved by any constant operation. Then, $\neq$ is pp-definable in Γ.*

▶ **Lemma 6.** *Let Γ be a non-trivial equality constraint language. Then, Γ pp-defines $=$ and $\neq$.*

Proof. Since Γ is non-trivial, in particular, Γ is not preserved by any constant operation. Thus, $\neq$ is pp-definable in Γ from Lemma 5.

Since Γ is non-trivial, there exists a satisfiable relation $R(x_1, \dots, x_k)$ pp-definable in Γ such that it is not satisfied by any *all-different* assignment β, where $\beta(x_i) \neq \beta(x_j)$ holds for every $i \neq j$. This means that for any satisfying assignment β for $R(x_1, \dots, x_k)$, $\beta(x_i) = \beta(x_j)$ holds for some $i \neq j$. As long as there is a pair of arguments (x_i, x_j) such that there is a satisfying assignment β with $\beta(x_i) \neq \beta(x_j)$, we add a constraint $(x_i \neq x_j)$ to R. Let R' be the resulting constraint. Note that R' is pp-definable in Γ as $\neq$ is pp-definable in Γ. Since R is not satisfied by the all-different assignment, R' must have some pair (x_i, x_j) such that we have not added the constraint $(x_i \neq x_j)$. Since R' becomes unsatisfiable if we add a constraint $(x_i \neq x_j)$ to R', x_i must be equal to x_j in any satisfying assignment of R'. Thus, the projection of R' to $\{x_i, x_j\}$ is the equality constraint. ◀

Combining Lemmas 4 and 6, the following holds.

▶ **Corollary 7.** *Let Γ be a non-trivial equality constraint language. Let R be a relation pp-definable in Γ. Then, $\mathit{CSP}(\Gamma \cup \{R\}) \leq_{\mathrm{RA}} \mathit{CSP}(\Gamma)$.*

Semidefinite Programming

We introduce an SDP relaxation BasicSDP. For an instance $\mathcal{I} = (V, \mathcal{C})$ of a standard CSP over the domain $[q]$, we want to find a collection of vectors $\{\boldsymbol{x}_{u,a}\}_{u \in V, a \in [q]}$ and a collection of probability distributions $\{\boldsymbol{\mu}_C\}_{C \in \mathcal{C}}$:

$$\begin{aligned}
&\max \mathop{\mathbf{E}}_{C \in \mathcal{C}} \Pr_{\beta \sim \boldsymbol{\mu}_C} [\beta \models C] \\
&\text{s.t.} \Pr_{\beta \sim \boldsymbol{\mu}_C} [\beta(u) = a, \beta(v) = b] = \langle \boldsymbol{x}_{u,a}, \boldsymbol{x}_{v,b} \rangle && \forall C \in \mathcal{C}, u, v \in V, a, b \in [q], \\
&\quad \Pr_{\beta \sim \boldsymbol{\mu}_C} [\beta(u) = a] = \langle \boldsymbol{x}_{u,a}, \boldsymbol{I} \rangle && \forall C \in \mathcal{C}, u \in V, a \in [q].
\end{aligned}$$

Here, $\boldsymbol{I}$ is any unit vector. Since $\boldsymbol{\mu}_C$ is a probability distribution, we implicitly impose $\langle \boldsymbol{x}_{u,a}, \boldsymbol{x}_{v,b} \rangle \geq 0$ and $\sum_a \boldsymbol{x}_{u,a} = \boldsymbol{I}$. See [22] for detailed explanation of BasicSDP. We define $\mathbf{sdp}(\mathcal{I})$ as the optimal SDP value of BasicSDP for $\mathcal{I}$. For TCSP, since we only need n values though the domain is $\mathbb{Q}$, we can write down BasicSDP as well. Guruswami et al. showed that, assuming UGC, BasicSDP gives a tight approximation ratio to Ordering CSP, which is a large subset of TCSP. The difference is that, in Ordering CSP, we only consider constraints that can be satisfied only when all variables have different values. However, it is almost direct to modify the argument to cover the whole TCSP:

▶ **Lemma 8** ([14])**.** *Let Γ be a temporal constraint language. Suppose that there is an instance $\mathcal{I}$ of $\mathit{CSP}(\Gamma)$ with $\mathbf{sdp}(\mathcal{I}) = c$ and $\mathbf{opt}(\mathcal{I}) = s$. Then, it is UG-Hard to compute $(c - \varepsilon, s + \varepsilon)$-approximation for $\mathit{CSP}(\Gamma)$ for any $\varepsilon > 0$.*

Let Γ be an equality constraint language and $\mathcal{I}$ be an instance of $\mathsf{CSP}(\Gamma)$. Then, $\mathbf{sdp}(\mathcal{I})$ is determined by $\sum_{a \in [q]} \langle \boldsymbol{x}_{u,a}, \boldsymbol{x}_{v,a} \rangle$ for $u, v \in V$. Thus, by letting $\boldsymbol{x}_u = \oplus_{a=1}^{q} \boldsymbol{x}_{u,a} := (\boldsymbol{x}_{u,1}, \dots, \boldsymbol{x}_{u,q})$, we can transform BasicSDP to the following SDP relaxation.

$$\begin{aligned}
&\max \mathop{\mathbf{E}}_{C \in \mathcal{C}} \Pr_{\beta \sim \boldsymbol{\mu}_C} [\beta \models C] \\
&\text{s.t.} \Pr_{\beta \sim \boldsymbol{\mu}_C} [\beta(u) = \beta(v)] = \langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle \quad \forall C \in \mathcal{C}, u, v \in V.
\end{aligned}$$

Again, we implicitly impose $\langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle \geq 0$ and $\|\boldsymbol{x}_u\|^2 = 1$. (Strictly speaking, the above formulation might be weaker than the original BasicSDP but suffices for our purpose.) Note that semidefinite programs can be solved within an additive error δ for any $\delta > 0$ in time polynomial in the size of an instance and $\log(1/\delta)$.

3 Qualitative Characterization

In this section, we prove Theorem 1, which is a "qualitative" characterization of robust approximability.

First we introduce well-known relations (See [7]).

- Betw is the ternary relation $\{(x, y, z) \in \mathbb{Q}^3 \mid (x < y < z) \vee (z < y < x)\}$.
- Cycl is the ternary relation $\{(x, y, z) \in \mathbb{Q}^3 \mid (x < y < z) \vee (y < z < x) \vee (z < x < y)\}$.
- Sep is the 4-ary relation $\{(x_1, y_1, x_2, y_2) \in \mathbb{Q}^4 \mid$ all distinct and the interval $[\min\{x_1, y_1\}, \max\{x_1, y_1\}]$ and the interval $[\min\{x_2, y_2\}, \max\{x_2, y_2\}]$ overlap$\}$.

Then, we use the following classification result.

▶ **Lemma 9** (Theorem 20 (and proof of Theorem 50) in [7]). *A temporal constraint language Γ satisfies at least one of the following:*

1. *Γ is trivial,*
2. *There is a pp-definition of $<$, Cycl, Betw, or Sep in Γ, or*
3. *Γ is an equality constraint language.*

For the first case, robust approximation is meaningless since every instance is satisfiable. As for the second case, robust approximation is hard from the following and Corollary 7.

▶ **Lemma 10** ([13, 7, 14]). *It is NP-Complete to solve CSP({Betw}), CSP({Cycl}), and CSP({Sep}), and it is UG-Hard to compute $(1-\varepsilon, 1/2+\varepsilon)$-approximation of CSP({<}) for any $\varepsilon > 0$.*

Now we focus on the third case, i. e., Γ is an equality constraint language. The following lemma gives the condition under which $\mathsf{CSP}(\Gamma)$ is solvable.

▶ **Lemma 11** (Theorem 1 and Lemma 8 in [6]). *Let Γ be a non-trivial trivial equality constraint language. The problem CSP(Γ) is polynomial-time solvable if Γ is Horn and NP-complete otherwise.*

We show the following robust approximation algorithm for Horn =-SAT in the next section.

▶ **Lemma 12.** *For any $\varepsilon > 0$, there is a polynomial-time $O(\frac{\log\log 1/\varepsilon}{\log 1/\varepsilon})$-robust approximation algorithm for Horn =-SAT.*

We finish the proof of Theorem 1 by combining Lemmas 9, 10, 11 and 12. Note that we combine two algorithms of Lemmas 11 and 12 to handle the cases $\varepsilon = 0$ and $\varepsilon > 0$.

3.1 Approximability of Horn =-SAT

Now we prove Lemma 12. For an integer k, let Γ_k be the equality constraint language that consists of Horn clauses of at most k literals. Note that every Horn formula is pp-definable in Γ_3 and Γ_3 contains the equality relation. Thus, from Lemma 4, it suffices to consider $\mathsf{CSP}(\Gamma_3)$ to prove Lemma 12. In this section, however, we give an $O(\frac{\log(k\log 1/\varepsilon)}{\log 1/\varepsilon})$-robust approximation algorithm for $\mathsf{CSP}(\Gamma_k)$ to see the dependency on k.

Let $\mathcal{I} = (V, \mathcal{C})$ be an instance of $\mathsf{CSP}(\Gamma_k)$. We let $\boldsymbol{y}_C = \Pr_{\beta \sim \boldsymbol{\mu}_C}[\beta \models C]$ in the BasicSDP. Then for each constraint $C \in \mathcal{C}$, we have a constraint of the form:

$$\boldsymbol{y}_C \leq \sum_{(u \neq v) \in C} (1 - \langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle) + \sum_{(u = v) \in C} \langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle.$$

Note that the latter sum contains at most one summand.

Let $\mathcal{I}$ be an instance with $\mathbf{opt}(\mathcal{I}) \geq 1 - \varepsilon$. Clearly $\mathbf{sdp}(\mathcal{I}) \geq 1 - \varepsilon$ holds, and it follows that $\boldsymbol{y}_C \geq 1 - \sqrt{\varepsilon}$ for at least a $(1 - \sqrt{\varepsilon})$-fraction of constraints. Then, we discard constraints C with $\boldsymbol{y}_C < 1 - \sqrt{\varepsilon}$. For simplicity of exposition, we assume that every constraint C satisfies $\boldsymbol{y}_C \geq 1 - \varepsilon$. This does not affect the final result since $O(\frac{\log(k \log 1/\varepsilon)}{\log 1/\varepsilon})$ remains the same by replacing ε with $\sqrt{\varepsilon}$. We also assume that $\varepsilon < 1/2$.

Our rounding scheme is as follows. Let $s \geq 1$ and $\delta = \delta(k, \varepsilon) \ll \varepsilon$ be parameters determined later. Let $h = \frac{2\sqrt{k}}{\delta} \log \frac{1}{\delta}$. We pick t from $\{h^0, h^1, h^2, \ldots, h^s\}$ uniformly at random. Then, we choose t random hyperplanes, which divides the entire space into 2^t cells. For each cell, we introduce a new value and assign the value to all variables in the cell. Note that the resulting assignment β only uses at most 2^t different values.

The following lemma is useful to analyze the performance of our algorithm.

▶ **Lemma 13.** *Let $\boldsymbol{x}, \boldsymbol{y}$ be unit vectors. The probability that two unit vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are in the same cell given by t random hyperplanes is $(1 - \frac{\angle(\boldsymbol{x}, \boldsymbol{y})}{\pi})^t$. In particular, the following hold.*

- *If $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \geq 1 - \varepsilon$, then the probability that $\boldsymbol{x}$ and $\boldsymbol{y}$ are in the same cell is $1 - O(t\sqrt{\varepsilon})$.*
- *If $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \leq 1 - \varepsilon$, then the probability that $\boldsymbol{x}$ and $\boldsymbol{y}$ are in the same cell is $\exp(-\Omega(t\sqrt{\varepsilon}))$.*

Proof. The first claim is obvious. If $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \geq 1 - \varepsilon$, then $\angle(\boldsymbol{x}, \boldsymbol{y}) \leq 2\sqrt{\varepsilon}$ holds, and it follows that $(1 - \frac{\angle(\boldsymbol{x}, \boldsymbol{y})}{\pi})^t \geq (1 - \frac{2\sqrt{\varepsilon}}{\pi})^t \geq 1 - \frac{2t\sqrt{\varepsilon}}{\pi}$. If $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \leq 1 - \varepsilon$, then $\angle(\boldsymbol{x}, \boldsymbol{y}) \geq \sqrt{2\varepsilon}$ holds, and it follows that $(1 - \frac{\angle(\boldsymbol{x}, \boldsymbol{y})}{\pi})^t \leq (1 - \frac{\sqrt{2\varepsilon}}{\pi})^t \leq \exp(-\frac{t\sqrt{2\varepsilon}}{\pi})$. ◀

The following three lemmas show that each kind of constraints is satisfied with high probability.

▶ **Lemma 14.** *Let C be a constraint of the form $(u = v)$. If $\boldsymbol{y}_C \geq 1 - \varepsilon$, then $\Pr[\beta \models C] = 1 - O(h^s\sqrt{\varepsilon})$.*

Proof. Since $\langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle \geq 1 - \varepsilon$, from Lemma 13, we have $\Pr[\beta \models C] = \mathbf{E}_t[1 - O(t\sqrt{\varepsilon})] = 1 - O(h^s\sqrt{\varepsilon})$. ◀

▶ **Lemma 15.** *Let C be a constraint of the form $(u_1 \neq v_1) \vee \cdots \vee (u_l \neq v_l)$. If $\boldsymbol{y}_C \geq 1 - \varepsilon$, then $\Pr[\beta \not\models C] = 1/s + \exp(-\Omega(h/\sqrt{2l}))$.*

Proof. We have $\sum_{i=1}^{l} \langle \boldsymbol{x}_{u_i}, \boldsymbol{x}_{v_i} \rangle \leq l - 1 + \varepsilon$. Thus, there exists some $i \in [l]$ with $\langle \boldsymbol{x}_{u_i}, \boldsymbol{x}_{v_i} \rangle \leq 1 - \frac{1-\varepsilon}{l}$. From Lemma 13, we have $\Pr[\beta \not\models C] = \mathbf{E}_t[\exp(-\Omega(t\sqrt{(1-\varepsilon)/l}))] = 1/s + \exp(-\Omega(h/\sqrt{2l}))$ (We used $\varepsilon < 1/2$). ◀

▶ **Lemma 16.** *Let C be a constraint of the form $(u_1 \neq v_1) \vee \cdots \vee (u_{l-1} \neq v_{l-1}) \vee (u_l = v_l)$. If $\boldsymbol{y}_C \geq 1 - \varepsilon$, then $\Pr[\beta \models C] = 1 - O(h^s\sqrt{\varepsilon}) - \delta - 1/s$.*

Proof. Let $\eta = 1 - \langle \boldsymbol{x}_{u_l}, \boldsymbol{x}_{v_l} \rangle$. Suppose that $\eta < 2\varepsilon$. Then, from Lemma 13, $\Pr[\beta \models C] \geq \Pr[\beta_{u_l} = \beta_{v_l}] = \mathbf{E}_t[1 - O(t\sqrt{\varepsilon})] = 1 - O(h^s\sqrt{\varepsilon})$

Suppose that $\eta \geq 2\varepsilon$. Then, there exists some $i \in [l-1]$ such that $\langle \boldsymbol{x}_{u_i}, \boldsymbol{x}_{v_i} \rangle \leq 1 - \frac{\eta - \varepsilon}{l-1} \leq 1 - \frac{\eta}{2l}$. Let $p_t^+ = \Pr[\beta \models (u_l = v_l) \mid t]$ and $p_t^- = \Pr[\beta \not\models (u_i \neq v_i) \mid t]$. We want to bound from above the number of t such that neither $p_t^+ \geq 1 - \delta$ nor $p_t^- \leq \delta$. We will choose δ so that

$p_1^+ \geq 1-\delta$. Let $t^* \in \{h^i\}_{i=0}^s$ be the smallest value such that $p_{t^*}^+ < 1-\delta$. If $t^* \geq h^s$, then we always have $p_t^+ \geq 1-\delta$ and we are done. Suppose $t^* < h^s$. Then, by choosing $s = \frac{1}{\delta}$ and $\delta = \frac{\log(k \log 1/\varepsilon)}{\log 1/\varepsilon}$, we have $p_{ht^*}^- \leq \delta$ as follows. From Lemma 13, $1-\delta > p_{t^*}^+ \geq 1 - \frac{2t^*\sqrt{\eta}}{\pi}$, hence $\delta < \frac{2t^*\sqrt{\eta}}{\pi}$. Multiplying h both sides and using the definition of h, we have $\log \frac{1}{\delta} < \frac{ht^*\sqrt{\eta/k}}{\pi}$. Again from Lemma 13,

$$p_{ht^*}^- \leq \exp(-\frac{ht^*\sqrt{\eta/l}}{\pi}) \leq \exp(-\frac{ht^*\sqrt{\eta/k}}{\pi}) < \exp(-\log\frac{1}{\delta}) = \delta.$$

Thus, all but one choice of t, $p_t^+ \geq 1-\delta$ or $p_t^- \leq \delta$ holds. Thus, $\Pr[\beta \models C] \geq \frac{1}{s} \cdot 0 + (1 - \frac{1}{s})(1-\delta) \geq 1 - \delta - \frac{1}{s}$. ◀

Proof of Lemma 12. If a constraint C is in Γ_k, then from Lemmas 14, 15, and 16, the probability that β does not satisfy C is at most $O(h^s\sqrt{\varepsilon}) + 1/s + \exp(-\Omega(h/\sqrt{2k})) + \delta$. From the choice of s, δ, we have $\Pr[\beta \models C] = 1 - O(\delta)$.

In general, if a constraint C is defined as a conjunction of w relations in Γ_k, we have $\Pr[\beta \models C] = 1 - O(w \cdot \delta)$ by union bound. ◀

4 Quantitative Characterization

In this section, we prove Theorem 2, which is a "quantitative" characterization of robust approximability. Item 4 is already proved in Lemma 12. Items 1, 2 and 3 will be proved in the following sections.

4.1 Inapproximability of Correlation Clustering

In this section, we prove Item 1 of Theorem 2. Since any non-trivial Γ pp-defines $=$ and $\neq$ from Lemma 6, it suffices to show the following.

▶ **Lemma 17.** *It is UG-Hard to compute $o(\sqrt{\varepsilon})$-robust approximation of CSP$(\{=, \neq\})$.*

We show a reduction from Max-CUT to CSP$(\{=, \neq\})$, then apply the following theorem.

▶ **Theorem 18** ([19]). *It is UG-Hard to compute $o(\sqrt{\varepsilon})$-robust approximation for Max-CUT.*

The reduction is as follows. Let a graph $G = (V, E)$ be an instance of Max-CUT. We construct a weighted graph $\widehat{G} = (\widehat{V}, E_= \cup E_\neq, W)$ as: (i) $\widehat{V} := \{v_i \mid v \in V, i \in \{0,1\}\}$. (ii) $E_= := \{(u_i, v_{1-i}) \mid (u,v) \in E, i \in \{0,1\}\}$. (iii) $E_\neq := \{(v_0, v_1) \mid v \in V\}$. (iv) $W : E_= \cup E_\neq \to [0,1]$ as $W(e) = \frac{1}{4|E|}$ if $e \in E_=$, $W(e) = \frac{d(v)}{4|E|}$ if $e = (v_0, v_1) \in E_\neq$. Here $d(v)$ denotes the degree of v in G. Note that $\sum_{e \in E_=} W(e) = \sum_{e \in E_\neq} W(e) = \frac{1}{2}$. We can regard $\widehat{G}$ as an instance of CSP$(\{=, \neq\})$, and the following two lemmas hold.

▶ **Lemma 19.** *If $\mathbf{opt}(G) \geq 1-\varepsilon$, then $\mathbf{opt}(\widehat{G}) \geq 1 - \varepsilon/2$.*

Proof. Let $l : V \to \{0,1\}$ be a labeling for G with $\mathbf{opt}(G) \geq 1-\varepsilon$. Define $\widehat{l} : \widehat{V} \to \{0,1\}$, a labeling of $\widehat{G}$, as: $\widehat{l}(v_0) = l(v)$ and $\widehat{l}(v_1) = 1 - l(v)$. Then, $\widehat{l}$ satisfies a $1-\varepsilon$ fraction of edges in $E_=$ and every edge in $E_\neq$. Therefore, $\mathbf{opt}(\widehat{G}) \geq (1-\varepsilon) \times 1/2 + 1/2 = 1 - \varepsilon/2$. ◀

▶ **Lemma 20.** *If $\mathbf{opt}(\widehat{G}) \geq 1-\varepsilon$, then $\mathbf{opt}(G) \geq 1 - 2\varepsilon$.*

Proof. First we show that if $\mathbf{opt}(\widehat{G}) = 1$, then $\mathbf{opt}(G) = 1$. Without loss of generality, we can assume that G is connected. An optimal cut $l : V \to \{0, 1\}$ is defined as follows. Pick an arbitrary vertex $v_0^* \in \widehat{V}$ and define $V_0 := \{v_0 \in \widehat{V} \mid v_0 \text{ is reachable from } v_0^* \text{ using only edges in } E_=\}$, and $l(v) = 0$ iff $v_0 \in V_0$. Note that if $(u, v) \in E$, then exactly one of u_0, v_0 is in V_0, thus, l is an optimal cut.

Now we assume $\mathbf{opt}(\widehat{G}) \geq 1 - \varepsilon$ and a labeling $\widehat{l} : \widehat{V} \to \{1, 2, \ldots, 2|V|\}$ is optimal. We say a pair of vertices (v_0, v_1) is *good* if $\widehat{l}(v_0) \neq \widehat{l}(v_1)$. Consider a subgraph $\widehat{G}'$ induced by good vertices from $\widehat{G}$. To obtain $\widehat{G}'$, we need to remove at most an ε fraction of edges from $\widehat{G}$. Thus, the total weight of satisfied edges is at least $1 - 2\varepsilon$ in $\widehat{G}'$. Let $\widehat{G}''$ be a subgraph obtained from $\widehat{G}'$ by deleting all unsatisfied edges. Then, we can construct a cut from the labeling of $\widehat{G}''$ so that $\mathbf{opt}(G) \geq 1 - 2\varepsilon$, by similar reasoning for the case of $\mathbf{opt}(\widehat{G}) = 1$. ◀

Combining Theorem 18 and Lemmas 19, 20, we complete the proof of Lemma 17.

4.2 Approximability of Negative =-SAT

In this section, we prove Item 2 of Theorem 2. For an integer k, let Γ_k be the equality constraint language consisting of negative clauses of at most k literals. Since every negative formula is pp-definable in Γ_k for some k, we consider $\mathsf{CSP}(\Gamma_k)$.

Given an instance $\mathcal{I} = (V, \mathcal{C})$ of $\mathsf{CSP}(\Gamma_k)$, let $\mathcal{C}_=$ be the set of constraints of the form $(u = v)$ and $\mathcal{C}_{\neq} = \mathcal{C} \setminus \mathcal{C}_=$. Then, we solve $\mathsf{BasicSDP}$. For each constraint $C \in \mathcal{C}$, we let $\boldsymbol{y}_C = \Pr_{\beta \sim \boldsymbol{\mu}_C}[\beta \models C]$. Then, we have:

$$\begin{aligned} &\boldsymbol{y}_C \leq \langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle \qquad \text{if } C \in \mathcal{C}_=, \\ &\boldsymbol{y}_C \leq \sum_{(u \neq v) \in C} (1 - \langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle) \quad \text{if } C \in \mathcal{C}_{\neq}. \end{aligned}$$

Our rounding scheme uses t random hyperplanes to define an assignment β as was the case for Horn =-SAT, but here we fix $t = 10\sqrt{k}\log(1/\varepsilon)$.

Proof of Item 2 of Theorem 2. We can safely assume that each constraint C satisfies $\boldsymbol{y}_C \geq 1/2$ (At most an $O(\varepsilon)$-fraction of constraints can satisfy $\boldsymbol{y}_C < 1/2$). For a constraint $C \in \mathcal{C}$, we set $\varepsilon_C = 1 - \boldsymbol{y}_C$.

We consider the loss caused by $\mathcal{C}_=$. From Lemma 13, if $\boldsymbol{y}_C \geq 1 - \delta$ for $C \in \mathcal{C}_=$, then $\Pr[\beta \models C] = 1 - O(\sqrt{k\delta}\log(1/\varepsilon))$. Thus, the total loss is proportional to

$$\begin{aligned} \frac{1}{m} \sum_{C \in \mathcal{C}_=} \sqrt{k\varepsilon_C}\log(1/\varepsilon) &\leq \frac{\sqrt{k}\log(1/\varepsilon)}{m} \sqrt{|\mathcal{C}_=| \sum_{C \in \mathcal{C}_=} \varepsilon_C} \\ &\leq \sqrt{k}\log(1/\varepsilon) \sqrt{\frac{1}{m} \sum_{C \in \mathcal{C}_=} \varepsilon_C} \leq \sqrt{k\varepsilon}\log(1/\varepsilon). \end{aligned}$$

The first inequality is by Cauchy-Schwartz.

We now turn to $\mathcal{C}_{\neq}$. Let $C \in \mathcal{C}_{\neq}$ be a constraint of l literals. Then, we have $\sum_{i=1}^{l} \langle \boldsymbol{x}_{u_i}, \boldsymbol{x}_{v_i} \rangle \leq l - 1 + 1/2 = l - 1/2$ from $\varepsilon_C \leq 1/2$. Thus, there exists some $i \in [l]$ with $\langle \boldsymbol{x}_{u_i}, \boldsymbol{x}_{v_i} \rangle \leq 1 - \frac{1}{2l}$. From Lemma 13, we have $\Pr[\beta \not\models C] = \exp(-\frac{t\sqrt{1/l}}{\pi}) = O(\sqrt{\varepsilon})$. Thus, the total loss is at most $\frac{1}{m} \sum_{C \in \mathcal{C}_{\neq}} O(\sqrt{\varepsilon}) = O(\sqrt{\varepsilon})$.

In summary, if a constraint C is in Γ_k, then the total loss is at most $O(\sqrt{k\varepsilon}\log(1/\varepsilon)) + O(\sqrt{\varepsilon}) = O(\sqrt{k\varepsilon}\log(1/\varepsilon))$. In general, if a constraint C is defined as a conjunction of w relations in Γ_k, the total loss is at most $O(w \cdot \sqrt{k\varepsilon}\log(1/\varepsilon))$ by union bound. ◀

4.3 Inapproximability of Non-Negative =-SAT

In this section, we prove Item 3 of Theorem 2. We introduce a relation $\mathrm{ODD}_3(x,y,z) = \{(x,y,z) \in \mathbb{Q}^3 \mid |\{x,y,z\}| = 1 \text{ or } 3\}$. We use the following fact.

▶ **Lemma 21** ([5]). *Let Γ be an equality constraint language such that Γ is not preserved by a constant operation and some relation $R \in \Gamma$ is not negative. Then, ODD_3 is pp-definable in Γ.*

From Corollary 7, Lemmas 5 and 21, it suffices to show the following inapproximability result.

▶ **Lemma 22.** *It is UG-Hard to compute $o(\frac{1}{\log 1/\varepsilon})$-robust approximation of $\mathsf{CSP}(\{\mathrm{ODD}_3, \neq\})$.*

We will give an instance $\mathcal{I}$ with $\mathbf{sdp}(\mathcal{I}) = 1-\varepsilon$ and $\mathbf{opt}(\mathcal{I}) = 1 - O(\frac{1}{\log 1/\varepsilon})$. Then, we have the desired result from Lemma 8. We borrow several ideas from [15], which shows that computing $o(\frac{1}{\log 1/\varepsilon})$-robust approximation of Horn SAT (over the Boolean domain) is UG-Hard.

Given a parameter k, our integrality gap instance $\mathcal{I} = (V, \mathcal{C})$ looks as follows.

$$\begin{aligned} \text{Variables} &: u_1, \dots, u_k, v_1, \dots, v_k \\ \text{Initial constraint} &: \mathrm{ODD}_3(u_1, u_1, v_1) \\ \text{Block } i\ (1 \le i \le k-1) &: \begin{cases} \mathrm{ODD}_3(u_i, v_i, u_{i+1}), \\ \mathrm{ODD}_3(u_i, v_i, v_{i+1}) \end{cases} \\ \text{Final constraint} &: (u_k \neq v_k) \end{aligned}$$

We intend to set $u_1 = v_1$ using the initial constraint and to set $u_i = v_i = u_{i+1} = v_{i+1}$ using Block i. Because of the final constraint, the instance $\mathcal{I}$ is unsatisfiable. Since $\mathcal{I}$ has $2k$ constraints, we have $\mathbf{opt}(\mathcal{I}) \le 1 - \frac{1}{2k}$.

Now we show that $\mathbf{sdp}(\mathcal{I}) \ge 1 - \frac{1}{\exp(k)}$. Suppose that we have fixed SDP vectors $\boldsymbol{x} = \{\boldsymbol{x}_v\}_{v \in V}$ in BasicSDP. Then, for each constraint $C \in \mathcal{C}$, the optimal probability distribution $\boldsymbol{\mu}_C$ is locally determined from $\boldsymbol{x}$. Thus, to construct a good SDP solution, we can concentrate on constructing good SDP vectors $\boldsymbol{x}$. We say that $\boldsymbol{x}$ *satisfies* a constraint C if there is a probability distribution $\boldsymbol{\mu}_C$ that is consistent with $\boldsymbol{x}$ such that $\Pr_{\beta \sim \boldsymbol{\mu}_C}[\beta \models C] = 1$.

For $\delta = \frac{1}{\exp(k)}$, our SDP vectors $\boldsymbol{x}$ will satisfy the initial constraint up to $1-\delta$ and completely satisfy Block i $(1 \le i \le k-1)$ and the final constraint. Since it is hard to construct all the SDP vectors at once, we make SDP vectors for each block first so that they agree with each other on some interface, and then we coalesce them together. The following definition and claim help us bring down the difficulty.

▶ **Definition 23** (partial SDP solution). Let $\mathcal{C}' \subseteq \mathcal{C}$ be a set of constraints. Then, SDP vectors $\{\boldsymbol{x}_v\}_{v \in V'}$ for $V' \subseteq V$ is said to be a partial SDP solution on $\mathcal{C}'$ if every constraint in $\mathcal{C}'$ is satisfied by $\boldsymbol{x}$. (In particular, $\boldsymbol{x}_v$ must be defined for every variable v that appears in $\mathcal{C}'$.)

An easy modification of Claim 7 of [15] gives the following.

▶ **Lemma 24** ([15]). *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{C}$ be two disjoint set of constraints. Let $\boldsymbol{x}^1 = \{\boldsymbol{x}^1_v\}_{v \in V_1}$ and $\boldsymbol{x}^2 = \{\boldsymbol{x}^2_v\}_{v \in V_2}$ be partial SDP solutions on $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. Suppose that, for all $u, v \in V_1 \cap V_2$, it holds that $\langle \boldsymbol{x}^1_u, \boldsymbol{x}^1_v \rangle = \langle \boldsymbol{x}^2_u, \boldsymbol{x}^2_v \rangle$. Then, there exists a partial SDP solution $\boldsymbol{y}$ on $\mathcal{C}_1 \cup \mathcal{C}_2$ that preserves inner products between vectors corresponding to variables in $\mathcal{C}_1 \cap \mathcal{C}_2$.*

Now we construct a partial SDP solution for each block.

▶ **Lemma 25.** *For any $0 \le \delta \le 1/2$ and $1 \le i \le k-1$, there exists a partial SDP solution $\{\boldsymbol{x}_{u_i}, \boldsymbol{x}_{v_i}, \boldsymbol{x}_{u_{i+1}}, \boldsymbol{x}_{v_{i+1}}\}$ to Block i such that*

$$\langle \boldsymbol{x}_{u_i}, \boldsymbol{x}_{v_i} \rangle = 1 - \delta \text{ and } \langle \boldsymbol{x}_{u_{i+1}}, \boldsymbol{x}_{v_{i+1}} \rangle = 1 - 2\delta.$$

Proof. Consider the following matrix whose columns and rows correspond to $\boldsymbol{x}_{u_i}$, $\boldsymbol{x}_{v_i}$, $\boldsymbol{x}_{u_{i+1}}$, $\boldsymbol{x}_{v_{i+1}}$ in this order and each element represents the inner product between corresponding vectors.

$$A = \begin{pmatrix} 1 & 1-\delta & 1-\delta & 1-\delta \\ 1-\delta & 1 & 1-\delta & 1-\delta \\ 1-\delta & 1-\delta & 1 & 1-2\delta \\ 1-\delta & 1-\delta & 1-2\delta & 1 \end{pmatrix}.$$

This matrix A satisfies the condition of the lemma, and we can construct a probability distribution satisfying Block i that is consistent to inner products determined by A. For example, for the constraint $\mathrm{ODD}_3(u_i, v_i, u_{i+1})$, we can use the probability distribution for which $|\{u_i, v_i, u_{i+1}\}| = 1$ with probability $1-\delta$ and $|\{u_i, v_i, u_{i+1}\}| = 3$ with probability δ.

To ensure there are vectors realizing the matrix A, we need to show that A is positive semidefinite. Let J be the all-one matrix. Then,

$$A = (1-2\delta)J + \delta \begin{pmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 0 \\ 1 & 1 & 0 & 2 \end{pmatrix}.$$

We can check that last matrix is positive semidefinite. Thus, A is a sum of semidefinite matrices and hence A is also positive semidefinite. ◀

▶ **Lemma 26.** $\mathbf{sdp}(\mathcal{I}) \ge 1 - \frac{1}{k2^{k+1}}$.

Proof. Let $\delta > 0$ be a sufficiently small value. By combining partial SDP solutions given by Lemma 25 iteratively using Lemma 24, we have an SDP solution $\boldsymbol{x} = \{\boldsymbol{x}_v\}_{v \in V}$ with the following property: it is a partial SDP solution for all constraints in Blocks 1 to $k-1$, and

$$\langle \boldsymbol{x}_{u_1}, \boldsymbol{x}_{v_1} \rangle = 1 - \delta, \quad \langle \boldsymbol{x}_{u_k}, \boldsymbol{x}_{v_k} \rangle = 1 - 2^k\delta.$$

Then, the loss from the initial constraint is δ, and the loss from the final constraint is $1 - 2^k\delta$. By choosing $\delta = 1/2^k$, the optimal SDP value is at least $1 - \frac{\delta}{2k} = 1 - \frac{1}{k2^{k+1}}$. ◀

Since $\mathbf{opt}(\mathcal{I}) \le 1 - \frac{1}{2k}$ whereas $\mathbf{sdp}(\mathcal{I}) \ge 1 - \frac{1}{k2^{k+1}}$, we have Lemma 22 from Lemma 8, which gives Item 3 of Theorem 2.

5 Robust Approximation of Horn =-SAT in Almost Linear Time

In this section, we show that we can solve BasicSDP for Horn =-SAT in almost linear time. Since rounding can be done in linear time, we can obtain an $O(\log\log(1/\varepsilon)/\log(1/\varepsilon))$-robust approximation for Horn =-SAT as well as an $O(\sqrt{\varepsilon}\log(1/\varepsilon))$-robust approximation for Negative =-SAT in almost linear time. Recall that Negative =-SAT is a special case of Horn =-SAT.

For a TCSP instance $\mathcal{I}$, let $\mathcal{I}_q$ be the instance whose domain is restricted to $[q]$ instead of $\mathbb{Q}$. The following lemma says that, if q is large enough, then the optimal value does not decrease much by using only q values.

▶ **Lemma 27.** *Let $\mathcal{I}$ be an instance of Horn =-SAT. Then,* $\mathbf{opt}(\mathcal{I}_q) \geq (1 - \frac{1}{q})\mathbf{opt}(\mathcal{I})$ *holds.*

Proof. Let $\beta^* : V \to \mathbb{Q}$ be the optimal assignment for $\mathcal{I}$. Let $\phi : \mathbb{Q} \to [q]$ be a random mapping. (We do not have to define the whole mapping explicitly as the size of the range of β^* is bounded by $|V|$.) Then, we construct q-valued β from β^* by setting $\beta_v = \phi(\beta^*_v)$. Let C be a constraint satisfied by β^*. If C is of the form $(u = v)$, then $\Pr[\beta \models C] = 1$. If C is of the form $\wedge_{i=1}^{k}(u_i = v_i) \to \mathbf{false}$ for some $k \geq 1$, then there exists some $i \in [k]$ such that $\beta^*(u_i) \neq \beta^*(v_i)$. Thus, $\Pr[\beta \models C] \geq 1 - \frac{1}{q}$. Finally, suppose C is of the form $\wedge_{i=1}^{k-1}(u_i = v_i) \to (u_k = v_k)$ for some $k \geq 1$. Then, $\beta^*(u_k) = \beta^*(v_k)$ holds or there exists some $i \in [k]$ such that $\beta^*(u_i) \neq \beta^*(v_i)$. From the same reasoning, $\Pr[\beta \models C] \geq 1 - \frac{1}{q}$ holds. ◀

For CSP over finite domains, it is known that an almost optimal SDP solution can be obtained in almost linear time as follows.

▶ **Lemma 28** ([25])**.** *Let $\mathcal{I} = (V, \mathcal{C})$ be a CSP instance on n variables over the domain $[q]$ with m constraints and maximum arity k. Suppose* $\mathbf{sdp}(\mathcal{I}) \geq \alpha$*. Then for every $\varepsilon > 0$, we can compute in time $m \cdot \mathrm{poly}(k^q/\varepsilon) \cdot \mathrm{poly}\log n$ an SDP solution of value at least $\alpha - \varepsilon$ that is feasible for a CSP instance $\mathcal{I}'$ obtained from $\mathcal{I}$ by discarding at most an ε-fraction of constraints.*

▶ **Lemma 29.** *Let $\mathcal{I} = (V, \mathcal{C})$ be an instance of Horn =-SAT of maximum arity k with* $\mathbf{opt}(\mathcal{I}) \geq \alpha$*. Then, we can compute in time $m \cdot \mathrm{poly}(k^{1/\varepsilon}/\varepsilon) \cdot \mathrm{poly}\log n$ an SDP solution of value at least $\alpha - O(\varepsilon)$.*

Proof. We set $q = 1/\varepsilon$. From Lemma 27, $\mathbf{opt}(\mathcal{I}_q) \geq \alpha - \varepsilon$. Using Lemma 28, we obtain a feasible SDP solution $\{\boldsymbol{x}_{u,a}\}_{u \in V, a \in [q]}$ of value at least $\alpha - O(\varepsilon)$. Here, the $O(\cdot)$ notation arises since we have discarded an ε-fraction of constraints from $\mathcal{I}_q$,

Now, we define $\boldsymbol{x}_u$ as $\oplus_{i=1}^{q} \boldsymbol{x}_{u,i}$ and claim $\{\boldsymbol{x}_u\}_{u \in V}$ is a good SDP solution for $\mathcal{I}$. The objective value does not change since $\langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle = \sum_{i \in [q]} \langle \boldsymbol{x}_{u,i}, \boldsymbol{x}_{v,i} \rangle$ and the objective value is only determined by these inner products. Moreover, constraints in BasicSDP are satisfied since $\|\boldsymbol{x}_u\|^2 = \sum_{i=1}^{q} \|\boldsymbol{x}_{u,i}\|^2 = 1$ and $\langle \boldsymbol{x}_u, \boldsymbol{x}_v \rangle = \sum_i \langle \boldsymbol{x}_{u,i}, \boldsymbol{x}_{v,i} \rangle \geq 0$. ◀

Combining the rounding method given in Sections 3 and 4, we have the following.

▶ **Corollary 30.** *For Horn =-SAT (resp., Negative =-SAT) of maximum arity k, In $m \cdot \mathrm{poly}(k^{1/\varepsilon}/\varepsilon) \cdot \mathrm{poly}\log n$, we can compute an $O(\frac{\log(k \log 1/\varepsilon)}{\log 1/\varepsilon})$-robust approximation (resp., an $O(\sqrt{\varepsilon}\log(1/\varepsilon)$-robust approximation).*

Acknowledgements. We would like to thank anonymous reviewers for their careful reading and comments on our paper. The comments helped us a lot in improving the paper.

References

1 Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004.
2 Libor Barto and Marcin Kozik. Robust satisfiability of constraint satisfaction problems. In *Proc. 44th Symp. on Theory of Computing Conference (STOC)*, pages 931–940, 2012.
3 Manuel Bodirsky. *Constraint satisfaction with infinite domains.* PhD thesis, Humboldt-Universität zu Berlin, 2004.
4 Manuel Bodirsky. Complexity classification in infinite-domain constraint satisfaction. *CoRR*, abs/1201.0856, 2012.
5 Manuel Bodirsky, Hubie Chen, and Michael Pinsker. The reducts of equality up to primitive positive interdefinability. *J. Symb. Log.*, 75(4):1249–1292, 2010.

6 Manuel Bodirsky and Jan Kára. The complexity of equality constraint languages. *Theory Comput. Syst.*, 43(2):136–158, 2008.

7 Manuel Bodirsky and Jan Kára. The complexity of temporal constraint satisfaction problems. *J. ACM*, 57(2), 2010.

8 Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *J. Comput. Syst. Sci.*, 71(3):360–383, 2005.

9 Nadia Creignou, Phokion G. Kolaitis, and Heribert Vollmer, editors. *Complexity of Constraints – An Overview of Current Research Themes [Result of a Dagstuhl Seminar]*, volume 5250 of *Lecture Notes in Computer Science*. Springer, 2008.

10 Víctor Dalmau and Andrei A. Krokhin. Robust satisfiability for CSPs: Hardness and algorithmic results. *TOCT*, 5(4):15, 2013.

11 Rina Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.

12 Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic snp and constraint satisfaction: A study through datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1998.

13 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman and Company, 1979.

14 Venkatesan Guruswami, Johan Håstad, Rajsekar Manokaran, Prasad Raghavendra, and Moses Charikar. Beating the random ordering is hard: Every ordering CSP is approximation resistant. *SIAM J. Comput.*, 40(3):878–914, 2011.

15 Venkatesan Guruswami and Yuan Zhou. Tight bounds on the approximability of almost-satisfiable Horn SAT and Exact Hitting Set. *Theory of Computing*, 8(1):239–267, 2012.

16 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

17 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing (STOC)*, pages 767–775, 2002.

18 Subhash Khot. On the unique games conjecture (invited survey). In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity (CCC)*, pages 99–121, 2010.

19 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.

20 Gábor Kun, Ryan O'Donnell, Suguru Tamaki, Yuichi Yoshida, and Yuan Zhou. Linear programming, width-1 CSPs, and robust satisfaction. In *Proceedings of the 3rd Innovations in Theoretical Computer Science (ITCS) conference*, pages 484–495, 2012.

21 Bernhard Nebel and Hans-Jürgen Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *J. ACM*, 42(1):43–66, 1995.

22 Prasad Raghavendra. *Approximating NP-hard problems: Efficient algorithms and their limits*. PhD thesis, University of Washington, 2009.

23 Francesca Rossi, Peter van Beek, and Toby Walsh, editors. *Handbook of Constraint Programming*. Foundations of Artificial Intelligence. Elsevier Science, 2006.

24 Thomas J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing (STOC)*, pages 216–226, 1978.

25 David Steurer. Fast SDP algorithms for constraint satisfaction problems. In *Proc. 21st Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 684–697, 2010.

26 Peter van Beek. Reasoning about qualitative temporal information. *Artif. Intell.*, 58(1–3):297–326, 1992.

27 Marc Vilain, Henry Kautz, and Peter van Beek. Constraint propagation algorithms for temporal reasoning: A revised report. In Daniel S. Weld and Johan de Kleer, editors, *Readings in Qualitative Reasoning about Physical Systems*, pages 373–381, 1989.

28 Uri Zwick. Finding almost-satisfying assignments. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 551–560, 1998.

Parity is Positively Useless*

Cenny Wenner

KTH – Royal Institute of Technology
`cenny@cwenner.net`

Abstract

We give the first examples of non-trivially positively-useless predicates subject only to $\mathsf{P} \neq \mathsf{NP}$. In particular, for every constraint function $Q : \{-1,1\}^4 \to \mathbb{R}$, we construct Contraint-Satisfaction-Problem (CSP) instances *without negations* which have value at least $1-\varepsilon$ when evaluted for the arity-four odd-parity predicate, yet it is NP-hard to find a solution with value significantly better than a random biased assignment when evaluated for Q. More generally, we show that all parities except one are positively useless.

Although we are not able to exhibit a single protocol producing hard instances when evaluated for every Q, we show that two protocols do the trick. The first protocol is the classical one used by Håstad with a twist. We extend the protocol to multilayered Label Cover and employ a particular distribution over layers in order to limit moments of table biases. The second protocol is a modification of Chan's multi-question protocol where queried tuples of Label Cover vertices are randomized in such a way that the tables can be seen as being independently sampled from a common distribution and in effect having identical expected biases. We believe that our techniques may prove useful in further analyzing the approximability of CSPs without negations.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Approximation hardness, approximation resistance, parity, usefulness, negations, monotone, constraint satisfaction problems, smoothness, multilayer, Label Cover

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.433

1 Introduction

We study the usefulness of boolean maximum *Constraint Satisfaction Problems* (CSPs). The Width-3 Parity Problem (Max E3-Even) and the Width-3 Satisfiability Problem (Max E3-SAT) are two canonical CSPs. In Max E3-Even resp. Max E3-SAT, an instance consists of a collection of constraints of the form $x'_{i_1} \oplus x'_{i_2} \oplus x'_{i_3}$ resp. $x'_{i_1} \vee x'_{i_2} \vee x'_{i_3}$ where $\oplus$ denotes `exclusive or` and x'_i is either a variable or its negation. A solution to an instance consists of an assignment to the variables and its value is the fraction of satisfied constraints. For the sake of analysis, varible domains is taken to be $\{-1,1\}$ where '1' is interpreted as `false` and '-1' as `true`. Similarly, constraints can be seen as a function from $\{-1,1\}^3$ to the real numbers applied to triples of variables, or their negations, and where a satisfying assignment of the variables is awarded the value 1. More generally, we define a CSP, denoted Max CSP(P), by specifying the *constraint function* $P : \{-1,1\}^k \to \mathbb{R}$ to be used instead of the 3-Even or 3-SAT constraints. The number of variables k which P acts on is called its width and if the range of P is contained in $\{0,1\}$, then P is called a *predicate*.

It turns out that for almost all predicates, it is NP-hard to find an assignment satisfying every constraint [19] and we turn our attention to *approximations*. We say that an algorithm

* Supported by ERC Advanced Grant 226203. This work was done in part while the author was visiting the Simons Institute for the Theory of Computing.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 433–448

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

A is a *(factor) c-approximation (algorithm)* if, given any instance, the value of the solution produced by A is within a factor c of the optimal value. One of the simplest approximation algorithms one could think of is to choose a uniform at random (u.a.r) assignment. This algorithm can be derandomized and, via linearity of expectation, constitutes an $\mathbf{E}[P]$ approximation. It turns out that for Max 3-Even and Max 3-SAT, this naive algorithm achieves the optimal constant-factor approximation assuming $\mathsf{P} \neq \mathsf{NP}$. More specifically, for every $\varepsilon > 0$, it is NP-hard to distinguish whether a Max 3-Even instance has value at least $1 - \varepsilon$ or at most $1/2 + \varepsilon$. Predicates, and CSPs, with the property that it is NP-hard to beat a random assignment are said to be *approximation resistant*. In a recent result by Chan [3], combined with a result by Håstad [11], it turns out that asymptotically a fraction one of predicates as k grows are in fact approximation resistant.

Inspired by stronger properties than approximability, Austrin and Håstad [1] introduced the concepts of *usefulness* and *uselessness*. A predicate P is said to be *(computationally) useful* for a constraint function Q if there exists a $\delta > 0$ and a polynomial-time algorithm which given instances of value at least $1-\delta$ evaluated for P, produces solutions of value at least $\mathbf{E}[Q]+\delta$ evaluated for Q. In other words, high-valued instances for P permit polynomial-time non-trivial solutions for Q. If there is no such algorithm, then P is said to be *useless for* Q. If P is useful for some (resp. no) constraint functions $Q : \{-1, 1\}^k \to \mathbb{R}$, then P is simply said to be *useful* (resp. *useless*). Note that assuming $\mathsf{P} \neq \mathsf{NP}$, uselessness can be established by showing NP-hardness. When $\mathsf{P} = \mathsf{NP}$, uselessness is instead essentially captured by the related definition called *information-theoretic uselessness*. However, information-theoretical usefulness is only of interest with respect to specific constraint functions as every constraint function is trivially information-theoretically useful for itself and hence useful given $\mathsf{P} = \mathsf{NP}$.

Assuming the Unique Games Conjecture (UGC) [14], Austrin and Håstad [1] gave a complete characterization of useless predicates: a predicate is useless if and only if there exists a pairwise-uniform distribution supported on $P^{-1}(1)$. This can be compared to the UGC-based result of Austrin and Mossel [2] which showed the same condition to be sufficient – but not necessary – for approximation resistance. This connection is not a coincidence; the general hardness of Austrin and Mossel inspired the study of Austrin and Håstad, and additionally, a predicate P is approximation resistant if and only if P is useless for itself. Consequently, every useless predicate is approximation resistant.

Similarly inspired to give a characterization of approximation resistance, Khot et al. [16] introduced the concept of *strong approximation resistance* where, for a predicate P, it is hard to not only find a solution with constant value greater than $\mathbf{E}[P]$ but it is hard to find any solutions with value significantly different than $\mathbf{E}[P]$, i.e., outside the range $\mathbf{E}P \pm o(1)$. Khot et al. gave sufficient and necessary condition of so called *strong approximation resistance* assuming the UGC. Curiously, if a predicate P is strongly approximation resistant, then P is useless for P as well as its complement $1 - P$. To the best of our understanding, the converse is however not known; uselessness of P for P resp. $1 - P$ can be establishing using distinct instances while strong approximation resistance demands that there are instances which are *simultaneously useless* for P and $1 - P$. Generalizing slightly, Austrin and Håstad named this stronger property *adaptive uselessness* for which we indeed have an equivalence.

Although we have a good understanding of the approximability of predicates assuming the UGC, until recently, little was known about approximation resistance conditioned only on $\mathsf{P} \neq \mathsf{NP}$. Using new instance constructions, Chan [3] showed the approximation resistance of all predicates P such that $P^{-1}(1)$ contains a group supporting a pairwise-uniform distribution. Again, by Håstad [11], the fraction of such predicates approaches one with the width. A caveat to this and other notable results is that the study is limited to CSP instances where

constraints act on both variables and their negations. This is the natural formulation for Max E3-SAT while it may be less reasonable for other problems. One well-known example of the distinction between allowing or not negations in a problem is the width-two `not-equal` predicate. If we permit negations, the predicate can encode both equality and inequality, and the resulting CSP is the Max 2-Lin-2 problem which is presently known to have a constant-approximation hardness of $\frac{11}{12} + \varepsilon$. When negations are not permitted, the problem corresponds to Max Cut which has a present constant-approximation hardness of $\frac{16}{17} + \varepsilon$ [10, 20]. Assuming the UGC, these two problems are in fact known to have the same approximability [15] but there are many other problems for which the hardness differs, such as Max E3-SAT where all constraints can be satisfied with an all-true assignment.

We denote by Max CSP$^+$(P) the restriction of Max CSP(P) where negations of variables are not automatically allowed and we call such instances *monotone* or "*without negations*". It turns out that a naive approximation algorithm for these problems can benefit from assigning variables the value 1 or -1 with different probabilities. The maximum expected value of such an algorithm is given by $\mathbf{E}^+P \stackrel{\text{def}}{=} \max_{b^*} \mathbf{E}_{\mathbf{x}_1,\ldots,\mathbf{x}_k \sim_{b^*} \{-1,1\}} [P(\mathbf{x}_1, \ldots, \mathbf{x}_k)]$, where $\mathbf{x} \sim_b \{-1, 1\}$ denotes drawing $\mathbf{x}$ with expectation b. When it is NP-hard to distinguish Max CSP$^+$(P) instances which have value at least $1 - \varepsilon$ from instances of value at most $\mathbf{E}^+P + \varepsilon$, we say that P is *positively approximation resistant*. Similarly, we say that P is *positively useful* for a constraint-function Q if there exist an $\varepsilon > 0$, there exists a polynomial-time algorithm which given monotone instances with value at least $1 - \varepsilon$ for P, produces solutions with value at least $\mathbf{E}^+Q + \varepsilon$ for Q. If P is not *positively useful* for Q, then it is *positively useless* for Q, and P is simply *positively useless* if it is so for every Q.

Assuming the UGC, Austrin and Håstad [1] gave a complete characterization also of positively-useless predicates under the UGC: P is positively useless if and only if $P^{-1}(1)$ supports a distribution where all biases are identical and (pairwise) correlations are non-negative and identical. For a more in-depth discussion and motivation of usefulness, we refer the reader to Austrin and Håstad [1].

Although UGC-based results are conditional, the conjecture has arguably contributed greatly in and outside the field of Approximability in the form of insights and techniques which have found applications also without the conjecture [17, 5], as well as promoting results which have subsequently been proven subject only to $\mathsf{P} \neq \mathsf{NP}$ [3, 9].

General conditional results have yet to be discovered for monotone CSPs. Despite the UGC implying that a fraction one of predicates for increasing width are positively approximation resistant, [1, 11], we are only aware of a handful of non-trivial natural predicates for which hardness is known subject only to $\mathsf{P} \neq \mathsf{NP}$. In particular, such results have been restricted to predicates where the optimal bias corresponds to balanced bits, i.e., $\mathbf{E}^+Q = \mathbf{E}\,Q$. One notable example is Max E4-Set Splitting, shown approximation resistant by Håstad [10], and generalized to greater widths by Guruswami [7]. Note that Max Ek-Set Splitting is the same problem as Max CSP$^+$(k-NAE) where k-NAE is the width-k predicate accepting heterogenous assignments. While there are a few results on the approximation resistance of monotone CSPs subject to $\mathsf{P} \neq \mathsf{NP}$, to the best of our knowledge, there were no known non-trivial positively-useless predicates prior to this work.

Organization. Our contributions are formally stated in Section 2, while Section 3 covers the analytical preliminaries, and Section 4 gives an overview to the two protocols and their analyses. Despite a generous page limit, we unfortunately only include the multilayered protocol, in Section 5 and its analysis, in Section 6. For the multiple-questions protocol, the generalization to other parities, and a proof of the hardness of the reduced-from Label Cover variant, we refer the reader to the full version.

2 Our Contributions

For every $k \in \mathbb{N}^{\geq 1}$, let k-Even (resp. k-Odd) be the width-k predicate satisfied by k-tuples containing an even (resp. odd) number of -1's. Certain predicates, such as k-Even, are known to be trivially positively useless because their monotone instances are satisfied by an all-1 or an all-(-1) assignment. In this work, we give the first examples of non-trivially positively-useless predicates subject only to $\mathsf{P} \neq \mathsf{NP}$.

	k-Even	k-Odd
width $k = 2$	triv. useless	pos. useful
$k \neq 2$ even	triv. useless	pos. useless
k odd	triv. useless	triv. useless

The positive usefulness of parities if $\mathsf{P} \neq \mathsf{NP}$.

▶ **Theorem 1.** *The predicate 4-Odd is positively useless iff* $\mathsf{P} \neq \mathsf{NP}$.

For presentational clarity, our analysis is chiefly concerned with $k = 4$ but the result generalizes to k-Odd for every even $k \geq 4$. We refer the reader to the full version for this generalization.

We note that also k-Odd is trivially satisfiable for odd k, and that for $k = 2$, Max CSP$^+$(k-Odd) is the Max Cut problem which is not positively useless since it has a non-trivial approximation by, e.g., Williamson and Goemans [6]. Consequently, the smallest parity candidate for positive uselessness is 4-Odd and more generally we show the following complete characterization of the positive usefulness of parities.

▶ **Theorem 2.** *Let P be even or odd parity of width $k \geq 1$. Then, P is positively useful if and only if P is 2-Odd or* $\mathsf{P} = \mathsf{NP}$.

A notable feature of our proof of Theorem 1 is that we exhibit two distinct protocols such that for every constraint function Q, one of the two protocols produce positively-useless instances for Q. While uselessness is implicit in many approximation-resistance proofs, to our knowledge, none of these results involve the combination of multiple protocols. This construction is somewhat non-intuitive, especially considering that a result by Austrin and Håstad [1] shows that a single protocol suffices for every positively-useless predicate assuming the UGC.

In the following sections we present a multilayered protocol and a multiple-questions protocol. The former establishes the uselessness of 4-Odd for every Q with positive highest Fourier coefficient while the second establishes uselessness for Q's with negative coefficient. Together they imply Theorem 1.

▶ **Lemma 3.** *The multilayered reduction $R_{\mathrm{ML},\gamma,s}$ from* LC *implies that 4-Odd is positively useless for Q whenever $\hat{Q}_{[4]} \geq 0$, subject to* $\mathsf{P} \neq \mathsf{NP}$.

▶ **Lemma 4.** *The multiple-questions reduction $R_{\mathrm{MQ},\gamma,p,M}$ from* LC *implies that 4-Odd is positively useless for Q whenever $\hat{Q}_{[4]} \leq 0$, subject to* $\mathsf{P} \neq \mathsf{NP}$.

3 Preliminaries

Random variables are for clarity denoted with bold font, as in $\mathbf{x}$, while vectors are denoted with overset arrows, as in $\vec{x}$. Vector multiplication is taken point wise. We let indexing of n elements range from 0 through $n-1$ and denote by $[n]$ the integer interval $\{0, \ldots, n-1\}$. For a statement S, the indicator $1\{\mathrm{S}\}$ is 1 if S is true and otherwise 0. For a function $\pi : R \to L$, we let $\pi(T)$ denote the image of π for the set $T \subseteq R$, and we use the notation $\pi_2(T)$ for the set of elements which T maps to an odd number of times.

For sets S and T we may denote their union $S \uplus T$ with the added condition that S and T are disjoint. We denote $\mathbf{x}$ being drawn uniformly at random (u.a.r.) from S with $\mathbf{x} \sim S$. When S consists of two real numbers $a < b$, typically -1 and 1, and $\mu \in [a, b]$, we let $\mathbf{x} \sim_\mu S$ indicate the distribution where $\mathbf{x}$ is chosen from S with expectation μ.

With respect to an implicit graph G, $N(v)$ signifies the neighborhood of a vertex v, the notation $u \sim v$ that the vertices u and v are neighbors, and for a sequence of vertex sets $U_1, \dots, U_k \subseteq V[G]$, the set of paths in G contained in $U_1 \times \cdots \times U_k$ is denoted $E(U_1, \dots, U_k)$. In particular, $E(U, W)$ is the set of edges between U and W.

Orthogonal Decompositions

For a function $f : \{-1,1\}^n \to \mathbb{R}$, we define $f(\vec{x}) = \sum_{S \subseteq [n]} \hat{f}_S \prod_{i \in S} x_i$ as the *Fourier expansion* of f where the quantity $\hat{f}_S$ is $\mathbf{E}_{\vec{\mathbf{x}}}\left[f(\vec{\mathbf{x}}) \prod_{i \in S} \mathbf{x}_i\right]$. We denote $x_S = \prod_{i \in S} x_i$. For every $S \subseteq [n]$, $\min f \le \hat{f}_S \le \max f$ and Parseval's Identity is of notable interest: $\sum \hat{f}_S^2 = \mathbf{E}\left[f^2\right]$. For a set Ω and function $f : \Omega^n \to \mathbb{R}$, we shall also use the *Efron-Stein decomposition* $\{f_S\}_{S \subseteq [n]}$ where $f_S(\vec{x}) \stackrel{\text{def}}{=} \sum_{T \subseteq S} (-1)^{|S \setminus T|} \mathbf{E}[f(\vec{\mathbf{y}}) \,|\, \vec{\mathbf{y}}_T = \vec{x}_T]$, satisfying $f = \sum f_S$; $f_S(\vec{x})$ only depends on $\{x_i\}_{i \in S}$; and whenever $T \setminus S \neq \emptyset$, $\mathbf{E}[f(\vec{\mathbf{x}}) \,|\, \vec{x}_T] = 0$. For a motivation of this decomposition, we refer the reader to Mossel [18].

▶ **Definition 5.** For any $0 \le \gamma \le 1$ and $f : \{-1,1\}^n \to \mathbb{R}$, the *noise operator* applied to f, $\mathrm{T}_{1-\gamma} f$, is defined as $\mathrm{T}_{1-\gamma} f(\vec{x}) \stackrel{\text{def}}{=} \mathbf{E}_{\vec{\mathbf{y}}}[f]$ where independently for each $i \in [n]$, $\mathbf{y}_i$ is set to x_i with probability $1 - \gamma$ and otherwise sampled uniformly at random from $\{-1, 1\}$.

For a function $f : \{-1,1\}^n \to \mathbb{R}$, the Fourier expansion of $\mathrm{T}_{1-\gamma} f$ is particularly simple: $\sum_S (1-\gamma)^{|S|} \hat{f}_S x_S$. Additionally, mean values are invariant of noise: $\mathbf{E}\left[\mathrm{T}_{1-\gamma} f\right] = \mathbf{E}\left[f\right]$.

3.1 Label Cover and Inapproximability

The Label Cover (LC) problem is a common starting point of strong inapproximability results. In particular, we will concern ourselves with a smooth multilayered variant which has an additional property which we call *path samplability*. Multilayered LC, or PCPs, were first introduced by Dinur et al. [4] for studying the approximability of Hypergraph Vertex Cover.

▶ **Definition 6.** An instance $(G, \{L_i\}_{[m_{\mathrm{LC}}]}, \{\pi^e\}_{E(G)})$ of the *m_{LC}-multilayered maximization problem* Label Cover consists of m_{LC} label sets $L_0, \dots, L_{m_{\mathrm{LC}}-1}$, and an m_{LC}-partite graph $G = (V_0 \uplus \cdots \uplus V_{m_{\mathrm{LC}}-1}, E)$ associating for every pair of vertex sets $V_a, V_b, a < b$; and edge $(\mathbf{u}, \mathbf{v}) \in E(V_a, V_b)$, a *projection* $\pi^{\mathbf{v},\mathbf{u}} : L_b \to L_a$. A solution to the instance consists of a labeling $\lambda : \biguplus V_a \to \biguplus L_a$ and its value is the maximum fraction of edges between two distinct vertex sets for which the labeling satisfies the associated projection:

$$\max_{a<b} \mathop{\mathbf{P}}_{(\mathbf{u},\mathbf{v}) \sim E(V_a,V_b)} \{\pi^{\mathbf{v},\mathbf{u}}(\lambda(\mathbf{v})) = \lambda(\mathbf{u})\}.$$

For a vertex $v \in \biguplus V_i$, we will also denote by $L(v)$ the label set L_i where V_i is the unique layer containing v. When restricting ourselves to the more common bipartite case, we denote $U = V_0; L = L_0; V = V_1$, and $R = L_1$. Since the label set L is typically smaller than R, we shall also refer to the vertices U resp. V as small- resp. large-side vertices. *Smoothness* of projections is a condition akin to but weaker than unique projections introduced by Khot [13] for analyzing approximate coloring of 3-colorable hypergraphs. We use the following definition of smoothness which is equivalent up to constants.

▶ **Definition 7.** An m_{LC}-multilayered LC instance is (J,ξ)-*smooth* when for every $0 \le a < b < m_{\mathrm{LC}}$, vertex $v \in V_b$, and set of at most J labels $S \subseteq L_b$, over a u.a.r. neighbor $\mathbf{u} \in V_a$ of v,

$$\mathop{\mathbf{P}}_{\mathbf{u}\sim V_a\cap N(v)}\{|\pi^{v,\mathbf{u}}(S)| \neq |S|\} \le \xi. \tag{1}$$

The most well-used bipartite LC constructions are known to be biregular and has the following sampling property: choosing a vertex v u.a.r. from either layer and a u.a.r. neighbor of v yields a u.a.r. edge in the graph. Our analysis requires a slightly stronger property: for every two LC layers $a < b$, choosing a u.a.r. path $\vec{p}$ from the first to the last layer and considering the vertices chosen in V_a and V_b yields a u.a.r. edge between V_a and V_b. We call this property path samplability and note that it is relatively easy to verify that, e.g., Khot's multilayered construction [13] satisfies this property. For a proof, we refer the reader to the full version.

▶ **Theorem 8.** *For every $J, \xi, \varepsilon_{LC} > 0$ and $m_{LC} \ge 2$, there exist path-samplable (J,ξ)-smooth m_{LC}-multilayered* LC *instances for which it is* NP*-hard to distinguish instances of value 1 from instances of value at most ε_{LC}.*

4 Protocols for Useless Instances

We have not been able to find a single protocol producing instances useless for every Q. However, by combining a new protocol with a multilayered-variant of a classical protocol, we argue that for every Q, at least one of these two protocols produces instances which are positively useless for Q.

Håstad's classical protocol [10] establishing the approximation resistance of 4-Odd with negations, samples u.a.r. an edge $(\mathbf{u}, \mathbf{v})$ from a bipartite LC instance, issues one query to an associated *table* $f^{\mathbf{u}}$ – a collection of variables seen as a function – and three to an associated table $f^{\mathbf{v}}$. With negations, each table can be assumed to have a balanced assignment of 1's and -1's via a folding trick. From this, one can argue that instances in the `no` case have value at most $\mathbf{E}\,[4\text{-Odd}] + \varepsilon$, implying approximation resistance, i.e., the uselessness of 4-Odd for 4-Odd. Without negations, the analysis does not carry through for the 4-Odd predicate for a simple reason. Since the protocol always issues one query to the first layer and three to the second, we could simply let the first layer consist entirely of -1's and the second entirely of 1's. Every constraint would then include one variable assigned -1 and three assigned 1, satisfying the 4-Odd predicate.[1]

Notably, Håstad's protocol does show the positive uselessness of 4-Odd for some constraint functions such as 4-Even. In turns out that a slight modification of the protocol suffices to extend this to every Q such that $\hat{Q}_{[4]} \ge 0$. The modification is to reduce from multilayered LC instances and sample tables from pairs of layers according to a particular distribution. The distribution ensures that the value analytically approximately corresponds to issuing randomly the three queries to either the lower or higher of the two layers, rather than always querying the higher layer thrice. This property ensures that whenever $\hat{Q}_{[4]} \ge 0$, the optimal table choices are essentially equally unbalanced. We note that a positive coefficient of $\hat{Q}_{[4]}$

[1] Note that one can show the positive approximation resistance of 4-Odd using a protocol which chooses a random $u \in U$, two neighbors $v_1, v_2 \sim u$ and issues two queries each to v_1 and v_2. A third possibility is the protocol which on a multilayered instance issues one query each to two distinct layers and two queries to a third. However, it turns out that there are still contraint functions Q for which one can deduce non-trivial solutions to all three of these protocols.

essentially means that even-parity assignments, such as zero, two, or four 1's; have a positive effect on the value while odd-parity assignments have a negative effect. This explanation agrees with the intuition of the protocol – if we can use the hardness of Label Cover to decouple the table queries, then we randomly query some table once and another thrice. We should then expect biased tables to only be able to achieve greater probabilities of odd-parity outcomes which is not beneficial.

Our second protocol is similar to Chan's protocol [3] for groups supporting pairwise independence. However, asking each table about several LC vertices serves a slightly different purpose in our construction than for Chan's hardness amplification. We construct constraints by sampling some large number of edges from a bipartite LC instance and independently for each edge and table, we ask the table about one of the two edge endpoints. This ensures that for any fixed set of edges, the four tables are independently drawn with replacement from a common distribution, implying that the four tables *have the same expected bias*. For technical reasons, asking each table about a single random endpoint is not sufficient. We pick many edges to ensure that each table is asked about both small-side and large-side endpoints, and almost always choose the large-side endpoint to avoid certain deleterious cases. Typically in the analysis of protocols showing approximation resistance, one argues that when reducing

from low-valued LC instances, the value of the protocol roughly corresponds to issuing independent queries to the respective chosen tables. Our construction does not have this property and the arguments can be highly correlated even for low-valued LC instances. However, we show that a certain kind of previously-known folding trick between arguments to a table still works and is enough to argue that the tables cannot coordinate better than independent biased assignments whenever $\hat{Q}_{[4]} \leq 0$.

5 The Multilayered Classical Protocol

We introduce the distribution over LC layers alluded to in Section 4. The distribution and lemmas are in particular taken from Guruswami et al., where the same was used to generate arguments for ordering problems [8]. In particular, we are using slight reformulations of their general distributions restricted to a domain of size $k = 2$.

For an integer $s > 0$, the distribution $\mathcal{D}_s$ could be seen as generating bit vectors by choosing a random suffix-length $0 \leq r < s$, a prefix $\vec{p}$ of length $s - r - 1$, and producing two random length-s strings $(\mathbf{a}, \mathbf{b})$, one with prefix $p \cdot (0)$ and one with prefix $p \cdot (1)$. This ensures that we always produce pairs for which $\mathbf{a} < \mathbf{b}$.

▶ **Definition 9.** For two integers $r, p \geq 0$, define $\mathcal{D}_{r,p}$ as the uniform distribution over $\{2^r \cdot p, \ldots, 2^r \cdot p + 2^r - 1\}$.

▶ **Definition 10** (Special case $k = 2$ of Definition 11.2, [8])**.** The distribution $\mathcal{D}_s$ is a distribution over pairs from $[2^s]$ defined as follows.

1. Pick a random $\mathbf{r}$ uniformly in $[s]$.
2. Pick a random $\mathbf{p}$ uniformly in $\left[2^{s-\mathbf{r}-1}\right]$.
3. Output $(\mathbf{a}, \mathbf{b})$ where $\mathbf{a}$ is sampled from $\mathcal{D}_{\mathbf{r}-1,2\mathbf{p}}$ and $\mathbf{b}$ from $\mathcal{D}_{\mathbf{r}-1,2\mathbf{p}+1}$.

The crucial property that we use this distribution for is that in spite of the distribution always generating pairs $(\mathbf{a}, \mathbf{b})$ for which $\mathbf{a} < \mathbf{b}$, when evaluated for discretized functions f and g, their expected product w.r.t. the distribution is close to an expectation over distributions where $\mathbf{a}$ and $\mathbf{b}$ are sampled independently.

In our analysis, we in particular use the following lemma which follows from discretizing the domain $[-1,1]$ and applying properties of $\mathcal{D}_s$ separately to f and g.

▶ **Lemma 11.** *Let $s \geq 0$ and consider $f, g : [2^s] \to [-1,1]$. Then for any integers $q > 0; k_1, k_2 \geq 0$,*

$$\left| \mathop{\mathbf{E}}_{(\mathbf{a},\mathbf{b})\sim\mathcal{D}_s} \left[f(\mathbf{a})^{k_1} g(\mathbf{b})^{k_2} \right] - \mathop{\mathbf{E}}_{\mathbf{r},\mathbf{p}} \left[\mathop{\mathbf{E}}_{\mathcal{D}_{\mathbf{r},\mathbf{p}}} \left[f(\mathbf{a})^{k_1} \right] \mathop{\mathbf{E}}_{\mathcal{D}_{\mathbf{r},\mathbf{p}}} \left[g(\mathbf{a})^{k_2} \right] \right] \right| \leq 2\frac{k_1+k_2}{q} + 8\sqrt{\frac{q}{s}}.$$

Proof. Deferred to Section 6. ◀

5.1 The Reduction

As is typical for LC reductions, we define a reduction to Max CSP$^+$ instances by specifying a probabilistic protocol where the weight of variable tuple in the produced instance corresponds to the probability that it is generated by the protocol.

▶ Procedure 12 (The Multi-Layered Protocol reduction $R_{\mathrm{ML},\gamma,s}$). Let $\gamma > 0; s \geq 2$ be arbitrary. The reduction $R_{\mathrm{ML},\gamma,s}$ from path-samplable $M = 2^s$-multilayered LC instances $(G, \{L_i\}_{[m_{\mathrm{LC}}]}, \{\pi^e\}_{E(G)})$ is defined with the following protocol.

1. Sample a pair of layer indices $(\mathbf{a}, \mathbf{b})$ from $\mathcal{D}_s$ and an edge $\mathbf{e} = (\mathbf{w_a}, \mathbf{w_b})$ from $E(V_\mathbf{a}, V_\mathbf{b})$.
2. Draw $\vec{\mathbf{x}}$ u.a.r. from $\{-1,1\}^{L_\mathbf{a}}$; $\vec{\mathbf{y}}^{(1)}$ and $\vec{\mathbf{y}}^{(2)}$ from $\{-1,1\}^{L_\mathbf{b}}$; and for each $j \in L_\mathbf{b}$, set $\mathbf{y}_j^{(3)} = -\mathbf{x}_{\pi^\mathbf{e}(j)} \mathbf{y}_j^{(1)} \mathbf{y}_j^{(2)}$.
3. For each $i \in L_\mathbf{a}$, resp. $j \in L_\mathbf{b}$, sample $\zeta_i^{(0)}, \zeta_j^{(1)}, \zeta_j^{(2)}$, and $\zeta_j^{(3)} \sim_{1-\gamma} \{-1,1\}$.
4. Output a random permutation of the tuple

$$\left(f^{(\mathbf{w_a})}(\vec{\zeta}^{(0)}\vec{\mathbf{x}}), f^{(\mathbf{w_b})}(\vec{\zeta}^{(1)}\vec{\mathbf{y}}^{(1)}), f^{(\mathbf{w_b})}(\vec{\zeta}^{(2)}\vec{\mathbf{y}}^{(2)}), f^{(\mathbf{w_b})}(\vec{\zeta}^{(3)}\vec{\mathbf{y}}^{(3)}) \right).$$

The distribution over pairs of vertices draws the pair $(\mathbf{a}, \mathbf{b}) \in [2^s]^2$ from $\mathcal{D}_s$ and outputs a random edge $(\mathbf{w_a}, \mathbf{w_b}) \in E(V_\mathbf{a}, V_\mathbf{b})$. Note that path samplability of LC instances implies that this distribution is equivalent to choosing a u.a.r. path $\vec{W} = (\mathbf{w}_0, \ldots, \mathbf{w}_{2^s-1}) \sim E(V_0, \ldots, V_{2^s-1})$ and a pair $(\mathbf{a}, \mathbf{b}) \sim \mathcal{D}_s$, generating the tuple $(\mathbf{w_a}, \mathbf{w_b})$.

▶ **Lemma 13** (Completeness). *Reducing from a satisfiable* LC *instance $\mathcal{I}$, the instance $R_{\mathrm{ML},\gamma,s}(\mathcal{I})$ produced by the Multilayered Protocol has value at least $1 - 4\gamma$ when evaluated for 4-Odd.*

This proof is standard. Consider a dictatorship assignment of an arbitrary satisfying labeling. If none of the coordinates used by the four dictators are noised, which happens with probability at least $1 - 4\gamma$, then the tuple of values of the queried tables equals $(x_1, x_2, x_3, -x_1x_2x_3)$ for some $x_1, x_2, x_3 \in \{-1,1\}$. For the 4-Odd predicate, such tuples are awarded value 1 and in effect, the expected value of the protocol is at least $1 - 4\gamma$.

▶ **Lemma 14** (Soundness). *Let $\xi, J, \varepsilon_{LC} > 0; s \geq 2$. Whenever Q satisfies $\hat{Q}_{[4]} \geq 0$ and the reduced-from 2^s-layered* LC *instance is path-samplable, (J,ξ)-smooth, and of value at most ε_{LC}, then the instance produced by $R_{\mathrm{ML},\gamma,s}$ has value at most*

$$\mathbf{E}^+ Q + 2\xi + 2(1-\gamma)^{3J} + \frac{2^9}{\sqrt[3]{s}} + \frac{\sqrt{\varepsilon_{LC}}}{\gamma}.$$

Proof. Deferred to Section 6. ◀

For appropriate choices of constants, the completeness and soundness of the reduction implies Theorem 3 – the uselessness for every Q such that $\hat{Q}_{[4]} \geq 0$.

6 Analysis of the Multilayered Protocol

In this section we complete the analysis of the multilayered protocol $R_{\mathrm{ML},\gamma,s}$. The analysis is split into five parts. First, we argue that the claimed completeness and soundness of $R_{\mathrm{ML},\gamma,s}$ indeed implies Theorem 3 – the positive uselessness of constraint-functions Q satisfying $\hat{Q}_{[4]} \geq 0$. Second, we give an outline of the soundness analysis of Q, split its proof into three lemmas, and argue that they imply the desired soundness. These three lemmas are subsequently proved, and respectively analyze the decoupling of table arguments, properties of the layer-distribution $\mathcal{D}_s$, and the bounding the value of decoupled evaluations of Q with averaged evaluations of Q.

6.1 Uselessness from the Protocol $R_{\mathrm{ML},\gamma,s}$

We argue that Theorem 3 in Section 2 follows from the supposed completeness and soundness of the multilayered protocol.

Proof of Theorem 3. The lemma follows if we can argue that, assuming $\mathsf{P} \neq \mathsf{NP}$, for arbitrary $\varepsilon' > 0$ and $Q' : \{-1,1\}^4 \to \mathbb{R}$ such that $\hat{Q'}_{[4]} \geq 0$, given a Max CSP$^+$ instance with value at least $1-\varepsilon$ when evaluated for 4-Odd, there is no polynomial-time algorithm to determine if the value is at least $\mathbf{E}^+ Q' + \varepsilon'$ evaluated for Q'. If $\min Q' = \max Q'$, the claim is trivial. Otherwise, we introduce the constraint function $Q = \frac{Q' - \min Q'}{\max Q' - \min Q'}$ and show the statement for $\varepsilon = \varepsilon'(\max Q' - \min Q')$. Note in particular that the codomain of Q' is $[0,1]$.

For the sake of contradiction, suppose that we had a polynomial-time algorithm A as above for some $\varepsilon > 0$. Let $\gamma = \varepsilon/4; \xi = \varepsilon/8; J \geq \ln_{1-\gamma}(\varepsilon/8); q \geq 2^{10}\varepsilon^{-1}; s \geq 2^6 q \varepsilon^{-2}$; and $\varepsilon_{\mathrm{LC}} = \gamma^2\varepsilon^2/16$. From Theorem 8, it is NP-hard to distinguish 2^s-multilayered LC instances of value 1 from path-samplable (J,ξ)-smooth instances of value at most $\varepsilon_{\mathrm{LC}}$. Given such an instance $\mathcal{I}$, consider running the supposed algorithm A on the $R_{\mathrm{ML},\gamma,s}(\mathcal{I})$. From Theorem 13, $R_{\mathrm{ML},\gamma,s}(\mathcal{I})$ has value at least $1-\varepsilon$ evaluated for 4-Odd. Consequently, A produces in polynomial time an instance of value at least $\mathbf{E}^+ Q + \varepsilon$ for Q by the choice of parameters, contradicting the assumption that $\mathsf{P} \neq \mathsf{NP}$. ◀

6.2 Properties of the Layer Distribution $\mathcal{D}_s$

We prove the property Theorem 11 of the layer-distribution $\mathcal{D}_s$.

▶ **Lemma 15** (Special case $k = 2$ of Lemma 11.3, [8]). *Let f be an arbitrary function from $[2^s]$ to a set S. When $\mathbf{r}, \mathbf{p}, \mathbf{a}, \mathbf{b}$ are chosen as in Theorem 10, for a random $\mathbf{j} \in \{0,1\}$,*

$$\sum_{\sigma \in S} \mathop{\mathbf{E}}_{\mathbf{r},\mathbf{p},\mathbf{j}} \left[\left| \mathop{\mathbf{P}}_{\mathbf{a} \sim \mathcal{D}_{\mathbf{r},\mathbf{p}}} \{f(\mathbf{a}) = \sigma\} - \mathop{\mathbf{P}}_{\mathbf{a} \sim \mathcal{D}_{\mathbf{r}-1,2\mathbf{p}+\mathbf{j}}} \{f(\mathbf{a}) = \sigma\} \right| \right] \leq \sqrt{\frac{|S|}{s}}.$$

Given two functions $f, g : [2^s] \to S$, Theorem 15 implies that $(f(\mathbf{a}), g(\mathbf{b}))$ where $(\mathbf{a}, \mathbf{b}) \sim \mathcal{D}_s$ is close in distribution to $(f(\mathbf{a}), g(\mathbf{b}))$ where $\mathbf{a}, \mathbf{b}$ are independently drawn from $\mathcal{D}_{\mathbf{r},\mathbf{p}}$ where $\mathbf{r} \sim [s], \mathbf{p} \sim \left[2^{s-\mathbf{r}-1}\right]$. In particular, although $\mathbf{a} < \mathbf{b}$ when drawn from $\mathcal{D}_s$, the same event only occurs with probability roughly $1/2$ in the latter case.

The decoupling Theorem 11 stated in Section 5 is a corollary of Theorem 15. We refer the interested reader to the full version for the proof.

6.3 Soundness of $R_{\mathrm{ML},\gamma,s}$

In this section, we prove the claimed soundness Theorem 14 of the $R_{\mathrm{ML},\gamma,s}$.

6.3.1 Notation

Let $\rho \stackrel{\text{def}}{=} 1-\gamma$ and for natural numbers x, y, let $x \nmid y$ denote that x does not divide y. Parametrized by an edge $e \in E$, we define the test distribution $\mathcal{T}^e = \mathcal{T}^{(u,v)}$ which independently samples $\{\zeta^{(t)}\}_{t=0}^3$ as random ρ-biased strings; draws $\vec{\mathbf{x}}, \vec{\mathbf{y}}^{(1)}, \vec{\mathbf{y}}^{(2)}$ uniformly at random; and for every $j \in L_b$, sets $\mathbf{y}_j^{(3)} = -\mathbf{x}_{\pi^{(e)}(j)} \mathbf{y}_j^{(1)} \mathbf{y}_j^{(2)}$. For notational clarity, we shall let the vertices $\mathbf{w_a}$, and $\mathbf{w_b}$ be implicit and denote $\mathbf{N} = N(\mathbf{w_b}) \cap V_{\mathbf{a}}$, $\pi = \pi^{(\mathbf{w_b}, \mathbf{w_a})}$, $\mathbf{f} = f^{\mathbf{w_a}}$, $\mathbf{g} = f^{(\mathbf{w_b})}$, and $\mathcal{T} = \mathcal{T}^{(\mathbf{w_a}, \mathbf{w_b})}$. For a sequence of vertices $\vec{w} \in V^{(0)} \times \cdots \times V^{(2^s-1)}$, introduce the function $\delta^{\vec{w}} : [2^s] \to [-1, 1]$ as a shorthand for the bias of the table in Layer a, i.e., $\delta^{\vec{w}}(a) \stackrel{\text{def}}{=} \mathbf{E}_{\vec{\mathbf{x}}} \left[f^{(w_a)}(\vec{\mathbf{x}}) \right]$. Finally, introduce the value of the protocol as

$$\text{Val} \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}=(\mathbf{w_a}, \mathbf{w_b}), \mathcal{T}} \left[Q\left(\mathbf{f}(\zeta^{(0)} \vec{\mathbf{x}}), \mathbf{g}(\zeta^{(1)} \vec{\mathbf{y}}^{(1)}), \mathbf{g}(\zeta^{(2)} \vec{\mathbf{y}}^{(2)}), \mathbf{g}(\zeta^{(3)} \vec{\mathbf{y}}^{(3)}) \right) \right],$$

and the *argument-decoupled value*,

$$\text{Val}_\perp \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}=(\mathbf{w_a}, \mathbf{w_b}), \mathcal{T}} \left[Q\left(\mathbf{E}\left[\mathbf{f}\right], \mathbf{E}\left[\mathbf{g}\right], \mathbf{E}\left[\mathbf{g}\right], \mathbf{E}\left[\mathbf{g}\right] \right) \right].$$

6.3.2 Outline

The proof outline is as follows. We argue that for low-valued LC instances, the value of the protocol is not significantly altered by drawing the arguments to the queried tables independently and uniformly at random. That is, the value of the protocol is approximately that of $Q\left(\delta^{\vec{\mathbf{w}}}(\mathbf{a}), \delta^{\vec{\mathbf{w}}}(\mathbf{b}), \delta^{\vec{\mathbf{w}}}(\mathbf{b}), \delta^{\vec{\mathbf{w}}}(\mathbf{b})\right)$ over random choices of the path $\vec{\mathbf{w}}$ and layer indices $\mathbf{a} < \mathbf{b}$. Exploiting the the distribution $\mathcal{D}_s$ of $(\mathbf{a}, \mathbf{b})$, the value is roughly equal when choosing $\mathbf{a}$ and $\mathbf{b}$ independently from a random distribution $\mathcal{D}_{\mathbf{r},\mathbf{p}}$. When $\mathbf{a}$ and $\mathbf{b}$ are drawn independently, we are able to compare the value to $Q\left(\frac{\delta^{\vec{\mathbf{w}}}(\mathbf{a})+\delta^{\vec{\mathbf{w}}}(\mathbf{b})}{2}, \ldots, \frac{\delta^{\vec{\mathbf{w}}}(\mathbf{a})+\delta^{\vec{\mathbf{w}}}(\mathbf{b})}{2}\right)$; which can be seen as drawing two tables and for each query picking a random point from one of the two tables with equal probability. In fact, this value and $Q\left(\delta^{\vec{\mathbf{w}}}(\mathbf{a}), \delta^{\vec{\mathbf{w}}}(\mathbf{b}), \delta^{\vec{\mathbf{w}}}(\mathbf{b}), \delta^{\vec{\mathbf{w}}}(\mathbf{b})\right)$ agree up to and including third moments. More carefully, the difference between the value of the protocol and the value of the random queries is given by $-\hat{Q}_{[4]} \cdot \left(\frac{\delta^{\vec{\mathbf{w}}}(\mathbf{a})-\delta^{\vec{\mathbf{w}}}(\mathbf{b})}{2}\right)^4$ which is non-positive for $\hat{Q}_{[4]} \geq 0$, in favor of the random queries. Consequently, the value of the protocol is approximately bounded from above by $Q\left(\frac{\delta^{\vec{\mathbf{w}}}(\mathbf{a})+\delta^{\vec{\mathbf{w}}}(\mathbf{b})}{2}, \ldots, \frac{\delta^{\vec{\mathbf{w}}}(\mathbf{a})+\delta^{\vec{\mathbf{w}}}(\mathbf{b})}{2}\right)$, which in turn is bounded by the maximum over independent biased bits: $\mathbf{E}^+ Q = \max_{b^*} Q(b^*, b^*, b^*, b^*)$.

6.3.3 Steps of the Soundness Analysis

The formal proof is divided into three steps with their respective lemmas. The first lemma argues that we can decouple table arguments when reducing from low-valued LC instances. The methods used to prove this lemma are standard and should come as no surprise to those familiar with the analysis of LC reductions.

▶ **Lemma 16.** *When the reduced-from* LC *instance is (J, ξ)-smooth and of value at most ε_{LC}, the value produced by the reduction $R_{\text{ML},\gamma,s}$ satisfies*

$$|\text{Val} - \text{Val}_\perp| \leq 2\rho^{3J} + 2\xi + \frac{\sqrt{\varepsilon_{LC}}}{\gamma}.$$

In the second step, we prove properties of the layer-distribution $\mathcal{D}_s$, Theorem 11, and that decoupled arguments drawn according to the $\mathcal{D}_s$ distribution has a simple expression in terms of the constraint-function Q.

▶ **Lemma 17.** *Using properties of the layer distribution $\mathcal{D}_s$, over some distribution of pairs $(\mathbf{x},\mathbf{y}) \in [-1,1]$, for every $s \geq 2$,*

$$\left|\mathrm{Val}_\perp - \mathbf{E}\left[\frac{Q(\mathbf{x},\mathbf{y},\mathbf{y},\mathbf{y}) + Q(\mathbf{x},\mathbf{x},\mathbf{x},\mathbf{y})}{2}\right]\right| \leq \frac{2^9}{\sqrt[3]{s}}.$$

Finally, we argue that regardless of the queried pair, when $\hat{Q}_{[4]} \geq 0$, evaluating Q using one random element of the pair once and the other thrice, is never beneficial in comparison to evaluating Q only with their average.

▶ **Lemma 18.** *For any $x, y \in \mathbb{R}$ and symmetric constraint-function Q s.t. $\hat{Q}_{[4]} \geq 0$,*

$$\frac{Q(x,y,y,y) + Q(x,x,x,y)}{2} \leq Q\left(\frac{x+y}{2}, \frac{x+y}{2}, \frac{x+y}{2}, \frac{x+y}{2}\right).$$

More generally, and recalling that $p \nmid q$ denotes that p does not divide q, Theorem 18 is a corollary of the following lemma.

▶ **Lemma 19.** *For any $x, y \in \mathbb{R}$, even k, and (multilinear extension of a) symmetric constraint function $Q : \{-1,1\}^k \to \mathbb{R}$ s.t. $\hat{Q}_{[k]} \geq 0$,*

$$\mathop{\mathbf{E}}_{\mathbf{k}_0 \sim \mathrm{Bin}(k,\frac{1}{2}) : 2 \nmid \mathbf{k}_0}\left[Q(\underbrace{x,\ldots,x}_{\mathbf{k}_0},\underbrace{y,\ldots,y}_{k-\mathbf{k}_0})\right] \leq Q\Big(\underbrace{\frac{x+y}{2},\ldots,\frac{x+y}{2}}_{k}\Big).$$

6.3.4 Soundness from Step Lemmas

Theorems 16 to 18 indeed implies the soundness of the reduction.

Proof of Theorem 14. Applying the three lemmas, over some distribution of pairs $(\mathbf{x},\mathbf{y}) \in [-1,1]$,

$$\mathrm{Val} \leq \mathbf{E}\left[Q\left(\frac{\mathbf{x}+\mathbf{y}}{2}, \frac{\mathbf{x}+\mathbf{y}}{2}, \frac{\mathbf{x}+\mathbf{y}}{2}, \frac{\mathbf{x}+\mathbf{y}}{2}\right)\right] + 2\xi + 2\rho^{3J} + \frac{\sqrt{\varepsilon_{\mathrm{LC}}}}{\gamma} + \frac{2^9}{\sqrt[3]{s}}.$$

Since the expectation of random variable is always bounded by its maximum, the first term is at most $\max_{b^*} Q(b^*,b^*,b^*,b^*) = \mathbf{E}^+ Q$ and hence corresponds to the stated bound. ◀

6.3.5 Decoupling Table Arguments

Proof of Theorem 16. The value of the protocol is given by

$$\mathrm{Val} \stackrel{\mathrm{def}}{=} \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}=(\mathbf{w_a},\mathbf{w_b}), \mathcal{T}}\left[Q\left(\mathbf{f}(\zeta^{(0)}\vec{\mathbf{x}}), \mathbf{g}(\zeta^{(1)}\vec{\mathbf{y}}^{(1)}), \mathbf{g}(\zeta^{(2)}\vec{\mathbf{y}}^{(2)}), \mathbf{g}(\zeta^{(3)}\vec{\mathbf{y}}^{(3)})\right)\right]. \tag{2}$$

Using the Fourier decomposition of Q and the definition of the noise operator,

$$(2) = \sum_{\Gamma \subseteq [4]} \hat{Q}_\Gamma \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathcal{T}}\left[\mathrm{T}_\rho f(\vec{\mathbf{x}})^{1\{0\in\Gamma\}} \prod_{t \in \Gamma\setminus\{0\}} \mathrm{T}_\rho g(\vec{\mathbf{y}}^{(t)})\right]. \tag{3}$$

We would like to compare this value to that of decoupled table arguments,

$$\begin{aligned}\mathrm{Val}_\perp &\stackrel{\mathrm{def}}{=} \mathop{\mathbf{E}}_{\vec{\mathbf{w}}, \mathcal{D}_s}\left[Q\left(\delta^{\vec{\mathbf{w}}}(\mathbf{a}), \delta^{\vec{\mathbf{w}}}(\mathbf{b}), \delta^{\vec{\mathbf{w}}}(\mathbf{b}), \delta^{\vec{\mathbf{w}}}(\mathbf{b})\right)\right] = \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}}\left[Q(\mathbf{E}\,\mathbf{f}, \mathbf{E}\,\mathbf{g}, \mathbf{E}\,\mathbf{g}, \mathbf{E}\,\mathbf{g})\right] \\ &= \sum_{\Gamma \subseteq [4]} \hat{Q}_\Gamma \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}}\left[(\mathbf{E}\,\mathbf{f})^{1\{0\in\Gamma\}} \prod_{t \in \Gamma\setminus\{0\}} \mathbf{E}\,\mathbf{g}\right]. \end{aligned} \tag{4}$$

Due to independence in the test distribution $\mathcal{T}$, all terms in the expansion of Q agree in Eq. (3) and Eq. (4), with exception of $\Gamma = \{1,2,3\}$ and $\Gamma = \{0,1,2,3\}$. Since we do not necessarily have balanced tables, our analysis cannot entirely follow Håstad's analysis [10] and instead we additionally employ smoothness similar to, e.g., Holmerin and Khot [12].

Note that if we can provide an absolute bound on the case $\Gamma = [4]$ for every pair of layers $a < b$ and every assignment to tables $\{f^{(w)}\}_{w \in V_a \cup V_b}$, then we also have a bound on the case $\Gamma = \{1,2,3\}$ by setting $f^{(w)} \equiv 1$ for all $w \in V_a$.

Hence for arbitrary fixed $a < b$ we proceed to bound the error for $\Gamma = [4]$:

$$\mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}=(\mathbf{w_a},\mathbf{w_b}), \mathcal{T}} \left[\mathbf{f}(\zeta^{(0)}\vec{\mathbf{x}}) \prod_{t=1}^{3} \mathbf{g}(\zeta^{(t)}\vec{\mathbf{y}}^{(t)}) \right]. \tag{5}$$

The following section shows that for LC instances of small value, the expectation Eq. (5) does not change much when the arguments are drawn independently. In particular, we bound

$$\left| \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}} \left[\mathop{\mathbf{E}}_{\mathcal{T}} \left[\mathbf{f}(\zeta^{(0)}\vec{\mathbf{x}}) \prod_{t=1}^{3} \mathbf{g}(\zeta^{(t)}\vec{\mathbf{y}}^{(t)}) \right] - \mathbf{E}\,\mathbf{f} \prod_{t=1}^{3} \mathbf{E}\,\mathbf{g} \right] \right|. \tag{6}$$

Taking the Fourier expansions of the functions, Eq. (6) equals

$$\left| \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}, \mathcal{T}} \left[\mathrm{T}_\rho \mathbf{f}(\vec{\mathbf{x}}) \prod_{t=1}^{3} \mathrm{T}_\rho \mathbf{g}(\vec{\mathbf{y}}^{(t)}) - \mathbf{E}\,\mathbf{f} \prod_{t=1}^{3} \mathbf{E}\,\mathbf{g} \right] \right| \tag{7}$$

$$= \left| \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}} \left[\sum_{\substack{S \subseteq L_\mathbf{a} \\ T_1,T_2,T_3 \subseteq L_b \\ S \cup T_1 \cup T_2 \cup T_3 \neq \emptyset}} \rho^{|S|} \hat{\mathbf{f}}_S \cdot \left(\prod_{t=1}^{3} \rho^{|T|} \hat{\mathbf{g}}_{T_t} \right) \mathop{\mathbf{E}}_{\mathcal{T}} \left[\left(\prod_{i \in S} \mathbf{x}_i \right) \left(\prod_{t=1}^{3} \prod_{j \in T_t} \mathbf{y}_j^{(t)} \right) \right] \right] \right|, \tag{8}$$

where we recall that $\mathbf{E}\,\mathbf{f} \prod_{t=1}^{3} \mathbf{E}\,\mathbf{g}$ expands to $\hat{\mathbf{f}}_\emptyset \prod_{t=1}^{3} \hat{\mathbf{g}}_\emptyset$, explaining the condition in the summation.

Using the definition of the test distribution $\mathcal{T}$, the inner expectation evaluates to 1 precisely when $S = \pi_2(T_3); T_1 = T_2 = T_3$ and otherwise to 0:

$$(8) = \left| \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}} \left[\sum_{\emptyset \neq T \subseteq L_\mathbf{b}} \rho^{|\pi_2(T)|+3|T|} \hat{\mathbf{f}}_{\pi_2(T)} \hat{\mathbf{g}}_T^3 \right] \right| \tag{9}$$

$$\leq \left| \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}} \left[\sum_{\substack{T \neq \emptyset \\ \pi_2(T) = \emptyset}} \rho^{3|T|} \hat{\mathbf{f}}_\emptyset \hat{\mathbf{g}}_T^3 \right] \right| + \left| \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}} \left[\sum_{\substack{T \neq \emptyset \\ \pi_2(T) \neq \emptyset}} \rho^{|\pi_2(T)|+3|T|} \hat{\mathbf{f}}_{\pi_2(T)} \hat{\mathbf{g}}_T^3 \right] \right|. \tag{10}$$

We bound separately the terms with $\pi_2(T)$ empty resp. non-empty.

Case $\pi_2(T)$ empty

Consider first the sum of terms with $\pi_2(T) = \emptyset$ and rewrite the expression as

$$\left| \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{w_a}, \mathbf{w_b}} \left[\sum_{T \neq \emptyset} (\hat{\mathbf{g}}_T^2) \hat{\mathbf{f}}_\emptyset \hat{\mathbf{g}}_T \rho^{3|T|} 1 \left\{ \pi_2^{(\mathbf{w_b},\mathbf{w_a})}(T) = \emptyset \right\} \right] \right|. \tag{11}$$

Using $\left|\hat{\mathbf{f}}_\emptyset \hat{\mathbf{g}}_T\right| \leq 1$,

$$(11) \leq \left| \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{w_b}} \left[\sum_{T \neq \emptyset} \hat{\mathbf{g}}_T^2 \rho^{3|T|} \mathop{\mathbf{P}}_{\mathbf{w_a} \sim \mathbf{N}} \left\{ \pi_2^{(\mathbf{w_b}, \mathbf{w_a})}(T) = \emptyset \right\} \right] \right|. \tag{12}$$

Since $\sum \hat{\mathbf{g}}_T^2 \leq 1$ via Parseval's identity, Eq. (12) is bounded by

$$\mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{w_b}} \left[\max_{T \neq \emptyset} \rho^{3|T|} \mathop{\mathbf{P}}_{\mathbf{w_a} \sim \mathbf{N}} \{ \pi_2^{\mathbf{w_b}, \mathbf{w_a}}(T) = \emptyset \} \right]. \tag{13}$$

When the reduced-from LC instance is (J, ξ)-smooth, the probability in the expression is by definition at most ξ whenever $|T| \leq J$. For larger sets, the factor $\rho^{3|T|}$ is at most ρ^{3J}. Consequently, we have a bound on the terms with $\pi_2(T) = \emptyset$,

$$(11) \leq (13) \leq \rho^{3J} + \xi. \tag{14}$$

Case $\pi_2(T)$ non-empty

We proceed to bound terms satisfying $\pi_2(T) \neq \emptyset$. Using the Cauchy-Schwarz Inequality and that $\sum \mathbf{g}_T^4 \leq 1$ via Parseval's identity, the second term in Eq. (9) is bounded by

$$\mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}} \left[\left(\sum_{T : \pi_2(T) \neq \emptyset} \rho^{2|\pi_2(T)| + 6|T|} \hat{\mathbf{f}}_{\pi_2(T)}^2 \hat{\mathbf{g}}_T^2 \right)^{\frac{1}{2}} \left(\sum_T \mathbf{g}_T^4 \right)^{\frac{1}{2}} \right], \tag{15}$$

$$\leq \max_{k>0} \{k \rho^k\} \mathop{\mathbf{E}}_{\mathcal{D}_s, \mathbf{e}} \left[\left(\sum_{T : \pi_2(T) \neq \emptyset} \frac{1}{|\pi_2(T)||T|} \hat{\mathbf{f}}_{\pi_2(T)}^2 \hat{\mathbf{g}}_T^2 \right)^{\frac{1}{2}} \right], \tag{16}$$

where the factor $\max_{k>0}\{k\rho^k\}$ is an upper bound on the ratio between $\left(\rho^{|\pi_2(T)|+|T|}\right)^{\frac{1}{2}}$ and $\left(\frac{1}{|\pi_2(T)||T|}\right)^{\frac{1}{2}}$. Note that an upper bound on $k\rho^k$ is in turn $\rho^1 + \cdots + \rho^k \leq (1-\rho)^{-1} = \gamma^{-1}$.

When Eq. (16) is significant, one expects to be able to derive a good labeling of the reduced-from LC instance. The labeling strategy we consider in showing this is the natural generalization of Håstad classical decoding. Given an assignment of the tables $\{f^{(w)}\}_{w \in \biguplus V_r}$, for every vertex $w \in \biguplus V_r$, choose the label i with probability $\sum_{S \ni i} \frac{1}{|S|} \widehat{f^{(w)}}_S^2$, and with the remaining probability choose an arbitrary label. This indeed defines probability distributions since, via Parseval's Identity, $\sum_i \sum_{S \ni i} \frac{1}{|S|} \widehat{f^{(w)}}_S^2 \leq \sum_S \widehat{f^{(w)}}_S^2 = 1$.

Between any two layers $a < b$, the labeling satisfies at least a fraction of constraints,

$$\mathop{\mathbf{E}}_{\mathbf{e}=(\mathbf{w}_a, \mathbf{w}_b) \sim E(V_a, V_b)} \left[\sum_{(i,j) \in \pi^{(\mathbf{e})}} \left(\sum_{S \ni i} \frac{1}{|S|} \widehat{\mathbf{f}}_S^2 \right) \left(\sum_{T \ni j} \frac{1}{|T|} \widehat{\mathbf{g}}_T^2 \right) \right]$$

$$= \mathop{\mathbf{E}}_{\mathbf{e}} \left[\sum_{S,T} \#\{(i,j) \in \pi \,:\, i \in S, j \in T\} \frac{\hat{\mathbf{f}}_S^2 \hat{\mathbf{g}}_T^2}{|S||T|} \right] \geq \mathop{\mathbf{E}}_{\mathbf{e}} \left[\sum_{T : \pi_2(T) \neq \emptyset} \frac{1}{|\pi_2(T)||T|} \hat{\mathbf{f}}_{\pi_2(T)}^2 \hat{\mathbf{g}}_T^2 \right]. \tag{17}$$

Since the value of a multilayered LC instance was defined as the maximum fraction of satisfied edges between two distinct layers, Eq. (17) is bounded from above by the value of the reduced-from LC instance, which in turn by assumption is at most $\varepsilon_{\mathrm{LC}}$.

Returning to the value of considered term,

$$(16) \leq \gamma^{-1} \mathop{\mathbf{E}}_{\mathcal{D}_s,\mathbf{e}} \left[\left(\sum_{T:\pi_2(T)\neq\emptyset} \frac{1}{|\pi_2(T)||T|} \hat{\mathbf{f}}^2_{\pi_2(T)} \hat{\mathbf{g}}^2_T \right)^{\frac{1}{2}} \right] \leq \frac{\sqrt{\varepsilon_{\mathrm{LC}}}}{\gamma}. \tag{18}$$

Completing the Argument Decoupling

Combining the bounds Eqs. (14) and (18), the error introduced by sampling the arguments independently for the term $\Gamma = [4]$ is $(5) \leq \rho^{3J} + \xi + \frac{\sqrt{\varepsilon_{\mathrm{LC}}}}{\gamma}$. From this, Eq. (14), and that $Q \to [0,1]$, we conclude

$$|\mathrm{Val} - \mathrm{Val}_\perp| \leq 2\rho^{3J} + 2\xi + \frac{\sqrt{\varepsilon_{\mathrm{LC}}}}{\gamma}. \tag{19}$$

◀

6.3.6 Decoupled Value to Symmetric Evaluation

We proceed to prove Theorem 17 – relating the decoupled value to an expectation of Q evaluated symmetrically for a random pair.

Proof of Theorem 17. Taking the Fourier expansion of the constraint function Q and recalling the definition of $\delta^{\vec{w}}$,

$$\mathrm{Val}_\perp = \sum \hat{Q}_\Gamma \mathop{\mathbf{E}}_{\mathcal{D}_s,\mathbf{e}} \left[\mathbf{E}\left[\mathbf{f}\right]^{1\{0\in\Gamma\}} \mathbf{E}\left[\mathbf{g}\right]^{|\Gamma\setminus\{0\}|} \right] = \sum \hat{Q}_\Gamma \mathop{\mathbf{E}}_{\vec{\mathbf{w}},\mathcal{D}_s} \left[\delta^{\vec{\mathbf{w}}}(\mathbf{a})^{1\{0\in\Gamma\}} \delta^{\vec{\mathbf{w}}}(\mathbf{b})^{|\Gamma\setminus\{0\}|} \right].$$

It is an issue for our analysis that the pair $(\mathbf{a},\mathbf{b}) \sim \mathcal{D}_s$ is always chosen so that $\mathbf{a} < \mathbf{b}$. However, due to the choice of the distribution $\mathcal{D}_s$, the value of two functions evaluated for $(\mathbf{a},\mathbf{b})$ is roughly unchanged when $\mathbf{a}$ and $\mathbf{b}$ are drawn independently from a random distribution. More specifically, using Theorem 11, for every choice of $\vec{w}$,

$$\left| \mathop{\mathbf{E}}_{\mathcal{D}_s} \left[\delta^{\vec{w}}(\mathbf{a})^{k_1} \delta^{\vec{w}}(\mathbf{b})^{k_2} \right] - \mathop{\mathbf{E}}_{\mathbf{r},\mathbf{p}} \left[\mathop{\mathbf{E}}_{\mathcal{D}_{\mathbf{r},\mathbf{p}}} \left[\delta^{\vec{w}}(\mathbf{a})^{k_1} \right] \mathop{\mathbf{E}}_{\mathcal{D}_{\mathbf{r},\mathbf{p}}} \left[\delta^{\vec{w}}(\mathbf{b})^{k_2} \right] \right] \right| \leq 2\frac{k_1+k_2}{q} + 8\sqrt{\frac{q}{s}}.$$

Applying the bound once for every $\Gamma \subseteq [4]$,

$$\left| \mathrm{Val}_\perp - \sum \hat{Q}_\Gamma \mathop{\mathbf{E}}_{\vec{\mathbf{w}},\mathbf{r},\mathbf{p}} \left[\mathop{\mathbf{E}}_{\mathcal{D}_{\mathbf{r},\mathbf{p}}} \left[\delta^{\vec{\mathbf{w}}}(\mathbf{a})^{1\{0\in\Gamma\}} \right] \mathop{\mathbf{E}}_{\mathcal{D}_{\mathbf{r},\mathbf{p}}} \left[\delta^{\vec{\mathbf{w}}}(\mathbf{b})^{|\Gamma\setminus\{0\}|} \right] \right] \right| \leq \min_q \left(\frac{2^7}{q} + 2^7\sqrt{\frac{q}{s}} \right).$$

Undoing the expansion of Q,

$$\left| \mathrm{Val}_\perp - \mathop{\mathbf{E}}_{\vec{\mathbf{w}},\mathbf{r},\mathbf{p}} \left[\mathop{\mathbf{E}}_{\mathbf{a},\mathbf{b}\sim\mathcal{D}_{\mathbf{r},\mathbf{p}}} \left[Q(\mathbf{x},\mathbf{y},\mathbf{y},\mathbf{y}) \right] \right] \right| \leq \min_q \left(\frac{2^7}{q} + 2^7\sqrt{\frac{q}{s}} \right),$$

$\mathbf{x} = \delta^{\mathbf{w}}(\mathbf{a})$ and $\mathbf{y} = \delta^{\mathbf{w}}(\mathbf{b})$. Since $\mathbf{a}$ and $\mathbf{b}$ are drawn independently from the same distribution, this indeed yields the desired bound for $s \geq 2$:

$$\left| \mathrm{Val}_\perp - \mathop{\mathbf{E}}_{\vec{\mathbf{w}},\mathbf{a},\mathbf{b}} \left[\frac{Q(\mathbf{x},\mathbf{x},\mathbf{x},\mathbf{y}) + Q(\mathbf{x},\mathbf{y},\mathbf{y},\mathbf{y})}{2} \right] \right| \leq \min_{q\in\mathbb{Z}_{\geq 1}} \left(\frac{2^7}{q} + 2^7\sqrt{\frac{q}{s}} \right) \leq \frac{2^9}{\sqrt[3]{s}}.$$

◀

6.3.7 Symmetric Evaluation to the Value of Querying Random Tables

We prove Theorem 18 with the more general Theorem 19.

Proof of Theorem 19. Fix arbitrary $x, y \geq \mathbb{R}$, k even, and symmetric $Q : \{-1,1\}^k \to \mathbb{R}$ such that $\hat{Q}_{[k]} \geq 0$. Since Q is symmetric, for every $m \in [k]$, we can introduce constants $\{\hat{Q}_m\}_{m\in[k]}$ such that for all $\Gamma \subseteq [k]$, $\hat{Q}_\Gamma = \hat{Q}_{|\Gamma|}$. We wish to upper-bound the expression

$$\begin{aligned}
&\mathop{\mathbf{E}}_{\mathbf{k}_0\sim\mathrm{Bin}(k,\frac{1}{2}) \mid 2\nmid \mathbf{k}_0} \left[Q(\underbrace{x,\ldots,x}_{\mathbf{k}_0},\underbrace{y,\ldots,y}_{k-\mathbf{k}_0})\right] \qquad (20)\\
&= \sum_{m=0}^{k} \hat{Q}_m \sum_{\Gamma\,:\,|\Gamma|=m} \mathop{\mathbf{E}}_{\mathbf{k}_0\sim\mathrm{Bin}(k,\frac{1}{2}) \mid 2\nmid \mathbf{k}_0} \left[x^{|\Gamma\cap[\mathbf{k}_0]|} y^{|\Gamma\cap\{\mathbf{k}_0,\ldots,k-1\}|}\right]
\end{aligned}$$

with

$$Q(\underbrace{\frac{x+y}{2},\ldots,\frac{x+y}{2}}_{k}) = \sum_{m=0}^{k} \hat{Q}_m \binom{k}{m} \left(\frac{x+y}{2}\right)^m. \qquad (21)$$

By symmetry,

$$(20) = \sum_{m=0}^{k} \hat{Q}_m \binom{k}{m} \mathop{\mathbf{E}}_{\mathbf{S}\subseteq[k]\,:\,2\nmid|S|} \left[x^{|\mathbf{S}\cap[m]|} y^{|\bar{\mathbf{S}}\cap[m]|}\right]. \qquad (22)$$

Since the elements in S are drawn $(k-1)$-wise independently, for $m < k$, the respective terms simply evaluate to $\hat{Q}_m\binom{k}{m}\left(\frac{x+y}{2}\right)^m$ which agree with the corresponding terms in Eq. (21).

We consider the term $m = k$ and use that k is even,

$$\begin{aligned}
&\hat{Q}_k \sum_{\Gamma\,:\,|\Gamma|=k} \mathop{\mathbf{E}}_{\mathbf{k}_0\sim\mathrm{Bin}(k,\frac{1}{2}) \mid 2\nmid \mathbf{k}_0} \left[x^{|\Gamma\cap[\mathbf{k}_0]|} y^{|\Gamma\cap\{\mathbf{k}_0,\ldots,k-1\}|}\right]\\
&= \hat{Q}_k \mathop{\mathbf{P}}_{\mathbf{S}\subseteq[k]} \{2\nmid|S|\}^{-1} \mathop{\mathbf{E}}_{\mathbf{S}\subseteq[k]} \left[x^{|\mathbf{S}|} y^{|\bar{\mathbf{S}}|} 1\{2\nmid|S|\}\right]\\
&= \hat{Q}_k \mathop{\mathbf{E}}_{\mathbf{S}\subseteq[k]} \left[x^{|\mathbf{S}|} y^{k-|\mathbf{S}|} \left(1-(-1)^{k-|\mathbf{S}|}\right)\right]\\
&= \hat{Q}_k \left(\left(\frac{x+y}{2}\right)^m - \left(\frac{x-y}{2}\right)^m\right) \leq \hat{Q}_k \left(\frac{x+y}{2}\right)^m,
\end{aligned}$$

where the final step uses that $\hat{Q}_k \geq 0$. The upper bound agrees with the term $m = k$ in Eq. (21), establishing the lemma. ◀

Acknowledgments. The author is grateful to Johan Håstad and Lukàš Polàček for many fruitful discussions and suggestions on this project, to anonymous referees for constructive comments, and to the hospitable Simons Institute for the Theory of Computing where part of this work was done.

References

1 Per Austrin and Johan Håstad. On the usefulness of predicates. *TOCT*, 5(1):1, 2013.

2 Per Austrin and Elchanan Mossel. Approximation resistant predicates from pairwise independence. *Computational Complexity*, 18(2):249–271, 2009.

3 Siu On Chan. Approximation resistance from pairwise independent subgroups. In *STOC*, pages 447–456, 2013.
4 Irit Dinur, Venkatesan Guruswami, Subhash Khot, and Oded Regev. A new multilayered pcp and the hardness of hypergraph vertex cover. *SIAM J. Comput.*, 34(5), 2005.
5 Uriel Feige, Guy Kindler, and Ryan O'Donnell. Understanding parallel repetition requires understanding foams. In *IEEE Conference on Computational Complexity*, 2007.
6 Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 1995.
7 Venkatesan Guruswami. Inapproximability results for set splitting and satisfiability problems with no mixed clauses. *Algorithmica*, 38(3):451–469, 2003.
8 Venkatesan Guruswami, Johan Håstad, Rajsekar Manokaran, Prasad Raghavendra, and Moses Charikar. Beating the random ordering is hard: Every ordering csp is approximation resistant. *SIAM J. Comput.*, 40(3):878–914, 2011.
9 Venkatesan Guruswami, Prasad Raghavendra, Rishi Saket, and Yi Wu. Bypassing ugc from some optimal geometric inapproximability results. In *SODA*, 2012.
10 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.
11 Johan Håstad. On the approximation resistance of a random predicate. *Computational Complexity*, 18(3):413–434, 2009.
12 Jonas Holmerin and Subhash Khot. A new pcp outer verifier with applications to homogeneous linear equations and max-bisection. In *STOC*, 2004.
13 Subhash Khot. Hardness results for coloring 3 -colorable 3 -uniform hypergraphs. In *FOCS*, pages 23–32, 2002.
14 Subhash Khot. On the power of unique 2-prover 1-round games. In *STOC*, 2002.
15 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), 2007.
16 Subhash Khot, Madhur Tulsiani, and Pratik Worah. A characterization of strong approximation resistance. *CoRR*, abs/1305.5500, 2013.
17 Subhash Khot and Nisheeth K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into l_1. In *FOCS*, 2005.
18 Elchanan Mossel. Gaussian bounds for noise correlation of functions. *Geometric and Functional Analysis*, 19, 2010.
19 Thomas J. Schaefer. The complexity of satisfiability problems. In *STOC*, 1978.
20 Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson. Gadgets, approximation, and linear programming. *SIAM J. Comput.*, 29(6):2074–2097, 2000.

The Condensation Phase Transition in Random Graph Coloring*

Victor Bapst[1], Amin Coja-Oghlan[1], Samuel Hetterich[1], Felicia Raßmann[1], and Dan Vilenchik[2]

1 **Mathematics Institute, Goethe University**
10 Robert Mayer St, Frankfurt 60325, Germany
{bapst, acoghlan, hetterich, rassmann}@math.uni-frankfurt.de

2 **Faculty of Mathematics & Computer Science, The Weizmann Institute**
Rehovot, Israel
dan.vilenchik@weizmann.ac.il

Abstract

Based on a non-rigorous formalism called the "cavity method", physicists have made intriguing predictions on phase transitions in discrete structures. One of the most remarkable ones is that in problems such as random k-SAT or random graph k-coloring, very shortly before the threshold for the existence of solutions there occurs another phase transition called *condensation* [Krzakala et al., PNAS 2007]. The existence of this phase transition seems to be intimately related to the difficulty of proving precise results on, e. g., the k-colorability threshold as well as to the performance of message passing algorithms. In random graph k-coloring, there is a precise conjecture as to the location of the condensation phase transition in terms of a distributional fixed point problem. In this paper we prove this conjecture, provided that k exceeds a certain constant k_0.

1998 ACM Subject Classification G.2.1 Combinatorics, G.2.2 Graph Theory

Keywords and phrases random graphs, graph coloring, phase transitions, message-passing algorithm

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.449

1 Introduction

Let $G(n,p)$ denote the random graph on the vertex set $V = \{1,\ldots,n\}$ obtained by connecting any two vertices with probability $p \in [0,1]$ independently. Throughout the paper, we are concerned with the setting that $p = d/n$ for a number $d > 0$ that remains fixed as $n \to \infty$. We say that $G(n,d/n)$ has a property with high probability *('w.h.p.') if its probability converges to 1 as $n \to \infty$.*

The study of random constraint satisfaction problems started with experimental work in the 1990s, which led to two hypotheses [5, 23]. First, that in problems such as random k-SAT or random graph coloring there is a *satisfiability threshold*, i. e., a critical "constraint density" below which the instance admits a solution and above which it does not w.h.p. Second, that this threshold is associated with the algorithmic "difficulty" of actually computing a solution, where "difficulty" has been quantified in various ways, albeit not in the formal sense

* The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement n. 278857–PTCC.

of computational complexity. These findings have led to a belief that random instances of k-SAT or graph k-colorability near the threshold for the existence of solutions are challenging algorithmic benchmarks, at the very least.

These two hypotheses have inspired theoretical work. Short of establishing the existence of an actual satisfiability threshold, Friedgut [13] and Achlioptas and Friedgut [1] proved that in random k-SAT and random graph k-coloring there exists a *sharp threshold sequence*. For instance, in the graph k-coloring problem, this is a sequence $d_{k-\mathrm{col}}(n)$ that marks the point where the probability of being k-colorable drops from 1 to 0.[1] The dependence on n allows for the possibility that this point might vary with the number of vertices, although this is broadly conjectured not to be the case. In fact, proving that $(d_{k-\mathrm{col}}(n))_{n\geq 1}$ converges to a single number $d_{k-\mathrm{col}}$ is a well-known open problem. So is determining the location of $d_{k-\mathrm{col}}(n)$ (or its limit), as [1] is a pure existence result.

In addition, inspired by predictions from statistical physics, the geometry of the set of solutions of random k-SAT or k-colorability instances has been investigated [2, 27]. The result is that at a certain point well before the satisfiability threshold the set of solutions shatters into a multitude of well-separated "clusters". Inside a typical cluster, all solutions agree on most of the variables/vertices, the so-called "frozen" ones. The average degree d at which these "frozen clusters" arise (roughly) matches the point up to which efficient algorithms provably find solutions.[2] Hence, on the one hand it is tempting to think that there is a connection between clustering and the computational "difficulty" of finding a solution [2, 27, 30]. On the other hand, physicists have suggested new *message passing algorithms* specifically to cope with a clustered geometry [4, 26]. A satisfactory analysis of these algorithms remains elusive.

Remarkably, the physics predictions are not merely circumstantial or experimental findings. They derive from a non-rigorous but systematic formalism called the *cavity method* [25]. This technique yields, among other things, a prediction as to the precise location of the k-SAT or k-colorability threshold. But perhaps even more remarkably, according to the cavity method shortly before the threshold for the existence of solutions there occurs another phase transition called *condensation* [20]. This phase transition marks a further change in the geometry of the solution space. While prior to the condensation phase transition each cluster contains only an exponentially small fraction of all solutions, thereafter a sub-exponential number of clusters contain a constant fraction of the entire set of solutions. As we will see in Section 3 below, condensation seems to hold the key to a variety of problems, including that of finding the k-colorability threshold and of analyzing message passing algorithms rigorously. More generally, the physicists' cavity method is extremely versatile. It has been used to put forward tantalizing conjectures in a variety of areas, including coding theory, probabilistic combinatorics, unsurprisingly, mathematical physics (see [25] for an overview) or, more recently, compressed sensing [19]. Hence the importance of providing a rigorous foundation for this technique.

2 Results

In this paper we prove that, indeed, a condensation phase transition occurs in random graph coloring, and that it occurs at the *precise* location predicted by the cavity method. This is

[1] Formally, for any $k \geq 3$ there is a sequence $(d_{k-\mathrm{col}}(n))_n$ such that for any fixed $\varepsilon > 0$, $G(n,p)$ is k-colorable w.h.p. if $p < (1-\varepsilon)d_{k-\mathrm{col}}(n)/n$, while $G(n,p)$ fails to be k-colorable w.h.p. if $p > (1+\varepsilon)d_{k-\mathrm{col}}(n)/n$.

[2] Actually the appearance of clusters does not quite match the appearance of frozen variables/vertices. For a more detailed explanation on the connection between clusters, frozen variables and computational hardness see [18, 21].

the first rigorous result to determine the exact location of the condensation transition in a model of this kind. Additionally, the proof yields a direct combinatorial explanation of how this phase transition comes about.

2.1 Catching a Sharp Threshold

To state the result, let us denote by $Z_k(G)$ the number of k-colorings of a graph G. We would like to study the "typical value" of $Z_k(G(n, d/n))$ in the limit as $n \to \infty$. As it turns out, the correct scaling of this quantity (to obtain a finite limit) is[3]

$$\Phi_k(d) \equiv \lim_{n\to\infty} \mathbb{E}[Z_k(G(n, d/n))^{1/n}].$$

In physics terminology, a "phase transition" is a point d_0 where the function $d \mapsto \Phi_k(d)$ is non-analytic. However, the limit $\Phi_k(d)$ is not currently known to exists for all d, k.[4] Hence, we need to tread carefully. For a given $k \geq 3$ we call $d_0 \in (0, \infty)$ *smooth* if there exists $\varepsilon > 0$ such that

- for any $d \in (d_0 - \varepsilon, d_0 + \varepsilon)$ the limit $\Phi_k(d)$ exists, and
- the map $d \in (d_0 - \varepsilon, d_0 + \varepsilon) \mapsto \Phi_k(d)$ has an expansion as an absolutely convergent power series around d_0.

If d_0 fails to be smooth, we say that a *phase transition* occurs at d_0.

For a smooth d_0 the sequence of random variables $(Z_k(G(n, d_0/n))^{1/n})_n$ converges to $\Phi_k(d_0)$ in probability. This follows from a concentration result for the number of k-colorings from [2]. Hence, $\Phi_k(d)$ really captures the "typical" value of $Z_k(G(n, d/n)$ (up to a factor of $\exp(o(n))$).

The above notion of "phase transition" is in line with the intuition held in combinatorics. For instance, the classical result of Erdős and Rényi [11] implies that the function that maps d to the limit as $n \to \infty$ of the expected fraction of vertices that belong to the largest component of $G(n, d/n)$ is non-analytic at $d = 1$. Similarly, if there actually is a sharp threshold $d_{k-\mathrm{col}}$ for k-colorability, then $d_{k-\mathrm{col}}$ is easily seen to be a phase transition in the above sense.[5]

As a next step, we state (an equivalent but slightly streamlined version of) the physics prediction from [22] as to the location of the condensation phase transition. As most predictions based on the "cavity method", this one comes in terms of a distributional fixed point problem. To be specific, let Ω be the set of probability measures on the set $[k] = \{1, \ldots, k\}$. We identify Ω with the k-simplex, i. e., the set of maps $\mu : [k] \to [0, 1]$ such that $\sum_{h=1}^{k} \mu(h) = 1$, equipped with the topology and Borel algebra induced by $\mathbb{R}^k$. Moreover, we define a map $\mathcal{B} : \bigcup_{\gamma=1}^{\infty} \Omega^\gamma \to \Omega$, $(\mu_1, \ldots, \mu_\gamma) \mapsto \mathcal{B}[\mu_1, \ldots, \mu_\gamma]$ by letting

$$\mathcal{B}[\mu_1, \ldots, \mu_\gamma](i) = \begin{cases} 1/k & \text{if } \sum_{h\in[k]} \prod_{j=1}^{\gamma} 1 - \mu_j(h) = 0, \\ \frac{\prod_{j=1}^{\gamma} 1-\mu_j(i)}{\sum_{h\in[k]} \prod_{j=1}^{\gamma} 1-\mu_j(h)} & \text{otherwise,} \end{cases} \quad \text{for any } i \in [k]\,. \tag{2.1}$$

[3] In the physics literature, one typically considers $n^{-1} \ln Z$ instead of $Z^{1/n}$, where Z is the so-called "partition function". We work with the nth root because our "partition function" Z_k may be equal to 0.

[4] It seems natural to conjecture that the limit $\Phi_k(d)$ exists for all d, k, but proving this might be difficult.In fact, the existence of the limit for all d, k would imply that $d_{k-\mathrm{col}}(n)$ converges.

[5] For $d < d_{k-\mathrm{col}}$, $G(n, d/n)$ has a k-coloring w.h.p., and thus the number of k-colorings is, in fact, exponentially large in n as there are $\Omega(n)$ isolated vertices w.h.p. Hence, if $\Phi_k(d)$ exists for $d < d_{k-\mathrm{col}}$, then $\Phi_k(d) > 0$. By contrast, for $d > d_{k-\mathrm{col}}$ the random graph $G(n, d/n)$ fails to be k-colorable w.h.p., and therefore $\Phi_k(d) = 0$. Thus, $\Phi_k(d)$ cannot be analytic at $d_{k-\mathrm{col}}$.

$$\begin{aligned}
\phi_{d,k}(\pi) &= \phi^e_{d,k}(\pi) + \frac{1}{k}\sum_{i\in[k]}\sum_{\gamma_1,\ldots,\gamma_k=0}^{\infty}\phi^v_{d,k}(\pi;i;\gamma_1,\ldots,\gamma_k)\prod_{h\in[k]}\left(\frac{d}{k-1}\right)^{\gamma_h}\frac{\exp(-d/(k-1))}{\gamma_h!}, \quad \text{where}\\
\phi^e_{d,k}(\pi) &= -\frac{d}{2k(k-1)}\sum_{h_1=1}^{k}\sum_{h_2\in[k]\setminus\{h_1\}}\int_{\Omega^2}\ln\left[1-\sum_{h\in[k]}\mu_1(h)\mu_2(h)\right]\bigotimes_{i=1}^{2}\mathrm{d}\pi_{h_i}(\mu_i), \qquad (2.4)\\
\phi^v_{d,k}(\pi;i;\gamma_1,\ldots,\gamma_k) &= \begin{cases}\ln k & \text{if } \sum_{i=1}^k\gamma_i=0,\\ \displaystyle\int_{\Omega^{\gamma_1+\cdots+\gamma_k}}\ln\left[\sum_{h=1}^{k}\prod_{h'\in[k]\setminus\{i\}}\prod_{j=1}^{\gamma_{h'}}1-\mu^{(j)}_{h'}(h)\right]\bigotimes_{h'\in[k]}\bigotimes_{j=1}^{\gamma_{h'}}\mathrm{d}\pi_{h'}(\mu^{(j)}_{h'}) & \text{if } \sum_{i=1}^k\gamma_i>0.\end{cases} \qquad (2.5)
\end{aligned}$$

Figure 1 The function $\phi_{d,k}$.

Further, let $\mathcal{P}$ be the set of all probability measures on Ω. For each $\mu\in\Omega$ let $\delta_\mu\in\mathcal{P}$ denote the Dirac measure that puts mass one on the single point μ. In particular, $\delta_{k^{-1}\mathbf{1}}\in\mathcal{P}$ signifies the measure that puts mass one on the uniform distribution $k^{-1}\mathbf{1}=(1/k,\ldots,1/k)$. For $\pi\in\mathcal{P}$ and $\gamma\geq 0$ let

$$Z_\gamma(\pi)=\sum_{h=1}^{k}\left(1-\int_\Omega\mu(h)\mathrm{d}\pi(\mu)\right)^\gamma. \qquad (2.2)$$

Further, define a map $\mathcal{F}_{d,k}:\mathcal{P}\to\mathcal{P}$, $\pi\mapsto\mathcal{F}_{d,k}[\pi]$ by letting

$$\begin{aligned}
\mathcal{F}_{d,k}[\pi] &= \exp(-d)\cdot\delta_{k^{-1}\mathbf{1}}\\
&\quad+\sum_{\gamma=1}^{\infty}\frac{\gamma^d\exp(-d)}{\gamma!\cdot Z_\gamma(\pi)}\int_{\Omega^\gamma}\left[\sum_{h=1}^{k}\prod_{j=1}^{\gamma}1-\mu_j(h)\right]\cdot\delta_{\mathcal{B}[\mu_1,\ldots,\mu_\gamma]}\bigotimes_{j=1}^{\gamma}\mathrm{d}\pi(\mu_j). \qquad (2.3)
\end{aligned}$$

Thus, in (2.3) we integrate a function with values in $\mathcal{P}$, viewed as a subset of the Banach space [6] of signed measures on Ω. The normalising term $Z_\gamma(\pi)$ ensures that $\mathcal{F}_{d,k}[\pi]$ really is a probability measure on Ω.

The main theorem is in terms of a fixed point of the map $\mathcal{F}_{d,k}$, i. e., a point $\pi^*\in\mathcal{P}$ such that $\mathcal{F}_{d,k}[\pi^*]=\pi^*$. In general, the map $\mathcal{F}_{d,k}$ has several fixed points. Hence, we need to single out the correct one. For $h\in[k]$ let $\delta_h\in\Omega$ denote the vector whose hth coordinate is one and whose other coordinates are 0 (i. e., the Dirac measure on h). We call a measure $\pi\in\mathcal{P}$ *frozen* if $\pi(\{\delta_1,\ldots,\delta_k\})\geq 2/3$; in words, the total probability mass concentrated on the k vertices of the simplex Ω is at least $2/3$.

As a final ingredient, we need a function $\phi_{d,k}:\mathcal{P}\to\mathbb{R}$. To streamline the notation, for $\pi\in\mathcal{P}$ and $h\in[k]$ we write π_h for the measure $\mathrm{d}\pi_h(\mu)=k\mu(h)\mathrm{d}\pi(\mu)$. With this notation, $\phi_{d,k}$ is defined in Figure 1. The integrals in (2.4) and (2.5) are well-defined because the set where the argument of the logarithm vanishes has measure zero.

▶ **Theorem 1.** *There exists a constant $k_0\geq 3$ such that for any $k\geq k_0$ the following holds. If $d\geq(2k-1)\ln k-2$, then $\mathcal{F}_{d,k}$ has precisely one frozen fixed point $\pi^*_{d,k}$. Further, the*

[6] To be completely explicit, the probability mass that a measurable set $A\subset\Omega$ carries under $\mathcal{F}_{d,k}[\pi]$ is

$$\mathcal{F}_{d,k}[\pi](A)=\exp(-d)\cdot\mathbf{1}_{\frac{1}{k}\mathbf{1}\in A}+\sum_{\gamma\geq 1}\frac{\gamma^d\exp(-d)}{\gamma!\cdot Z_\gamma(\pi)}\int\left[\sum_{h=1}^{k}\prod_{j=1}^{\gamma}1-\mu_j(h)\right]\cdot\mathbf{1}_{\mathcal{B}[\mu_1,\ldots,\mu_\gamma]\in A}\bigotimes_{j=1}^{\gamma}\mathrm{d}\pi(\mu_j),$$

where $\mathbf{1}_{\nu\in A}=1$ if $\nu\in A$ and $\mathbf{1}_{\nu\in A}=0$ otherwise.

function

$$\Sigma_k : d \mapsto \ln k + \frac{d}{2}\ln(1-1/k) - \phi_{d,k}(\pi^*_{d,k}) \tag{2.6}$$

has a unique zero $d_{k,\mathrm{cond}}$ *in the interval* $[(2k-1)\ln k - 2, (2k-1)\ln k - 1]$. *For this number* $d_{k,\mathrm{cond}}$ *the following three statments hold.*

(i) *Any* $0 < d < d_{k,\mathrm{cond}}$ *is smooth and* $\Phi_k(d) = k \cdot (1-1/k)^{d/2}$.

(ii) *There occurs a phase transition at* $d_{k,\mathrm{cond}}$.

(iii) *If* $d > d_{k,\mathrm{cond}}$, *then*

$$\limsup_{n\to\infty} \mathbb{E}[Z_k(G(n,d/n))^{1/n}] < k \cdot (1-1/k)^{d/2}.$$

Thus, if d *is smooth, then* $\Phi_k(d) < k \cdot (1-1/k)^{d/2}$.

The key strength of Theorem 1 and the main achievement of this work is that we identify the *precise* location of the phase transition. In particular, the result $d_{k,\mathrm{cond}}$ is one number rather than a "sharp threshold sequence" that might vary with n. Admittedly, this precise answer is not exactly a simple one. But that seems unsurprising, given the intricate combinatorics of the random graph coloring problem. That said, the proof of Theorem 1 will illuminate matters. For instance, the fixed point $\pi^*_{d,k}$ turns out to have a nice combinatorial interpretation and, perhaps surprisingly, $\pi^*_{d,k}$ emerges to be a *discrete* probability distribution.

The above formulas are derived systematically via the cavity method [25]. For instance, the functional $\phi_{d,k}$ is a special case of a general formula, the so-called "Bethe free entropy". Moreover, the map $\mathcal{B}$ is the distributional version of the "Belief Propagation" operator. In effect, the predictions as to the condensation phase transitions in other problems look very similar to the above. Consequently, it can be expected that the proof technique developed in the present work carries over to many other problems.

While the main point of Theorem 1 is that it gives an exact answer, it is not difficult to obtain a simple asymptotic expansion of $d_{k,\mathrm{cond}}$ in the limit of large k. Namely, $d_{k,\mathrm{cond}} = (2k-1)\ln k - 2\ln 2 + \varepsilon_k$, where $\varepsilon_k \to 0$ as $k \to \infty$. This asymptotic formula was obtained in [8] by means of a *much* simpler argument than the one developed in the present paper. However, this simpler argument does not quite get to the bottom of the combinatorics behind the condensation phase transition.

2.2 The Cluster Size

The proof of Theorem 1 allows us to formalise the physicists' notion that as d tends to $d_{k,\mathrm{cond}}$, the cluster size approaches the total number of k-colorings. Of course, we need to formalise what we mean by "clusters" first. Thus, let G be a graph on n vertices. If σ, τ are k-colorings of G, we define their *overlap* as the $k \times k$-matrix $\rho(\sigma,\tau) = (\rho_{ij}(\sigma,\tau))_{i,j\in[k]}$ with entries

$$\rho_{ij}(\sigma,\tau) = \frac{|\sigma^{-1}(i) \cap \tau^{-1}(j)|}{n},$$

i. e., $\rho_{ij}(\sigma,\tau)$ is the fraction of vertices colored i under σ and j under τ. Now, define the *cluster* of σ in G as

$$\mathcal{C}(G,\sigma) = \{\tau : \tau \text{ is a } k\text{-coloring of } G \text{ and } \rho_{ii}(\sigma,\tau) \geq 0.51/k \text{ for all } i \in [k]\}. \tag{2.7}$$

Suppose that σ, τ are such that $|\sigma^{-1}(i)|, |\tau^{-1}(i)| \sim n/k$ for all $i \in [k]$; most k-colorings of $G(n,d/n)$ have this property w.h.p. [1, 7]. Then $\tau \in \mathcal{C}(G,\sigma)$ means that a little over 50% of the vertices with color i under σ also have color i under τ.

▶ **Corollary 2.** *With the notation and assumptions of Theorem 1, the function Σ_k is continuous, strictly positive and monotonically decreasing on $((2k-1)\ln k - 2, d_{k,\mathrm{cond}})$, and $\lim_{d\to d_{k,\mathrm{cond}}} \Sigma_k(d) = 0$. Further, given that $G(n,d/n)$ is k-colorable, let $\boldsymbol{\tau}$ be a uniformly random k-coloring of this random graph. Then*

$$\lim_{\varepsilon\searrow 0}\lim_{n\to\infty} \mathbb{P}\left[\frac{1}{n}\ln\frac{|\mathcal{C}(G(n,d/n),\boldsymbol{\tau})|}{Z_k(G(n,d/n))} \le \Sigma_k(d)+\varepsilon \,\middle|\, \chi(G(n,d/n))\le k\right] = 1, \quad \textit{and}$$
$$\lim_{\varepsilon\searrow 0}\limsup_{n\to\infty} \mathbb{P}\left[\frac{1}{n}\ln\frac{|\mathcal{C}(G(n,d/n),\boldsymbol{\tau})|}{Z_k(G(n,d/n))} \ge \Sigma_k(d)-\varepsilon \,\middle|\, \chi(G(n,d/n))\le k\right] > 0.$$

We observe that our conditioning on the chromatic number $\chi(G(n,d/n))$ being at most k is necessary to speak of a random k-coloring $\boldsymbol{\tau}$ but otherwise harmless. For the first part of Theorem 1 implies that $G(n,d/n)$ is k-colorable w.h.p. for any $d < d_{k,\mathrm{cond}}$. Indeed, if $d < d_{k,\mathrm{cond}}$, then $\Phi_k(d) = k(1-1/k)^{d/2} > 0$ and thus $Z_k(G(n,d/n))^{1/n} > 0$ w.h.p. because $(Z_k(G(n,d/n))^{1/n})$ converges to $\Phi_k(d)$ in probability.

In words, Corollary 2 states that there is a certain function $\Sigma_k > 0$ such that the total number of k-colorings exceeds the number of k-colorings in the cluster of a randomly chosen k-coloring by at least a factor of $\exp[n(\Sigma_k(d)+o(1))]$ w.h.p. However, as d approaches $d_{k,\mathrm{cond}}$, $\Sigma_k(d)$ tends to 0, and with a non-vanishing probability the gap between the total number of k-colorings and the size of a single cluster is upper-bounded by $\exp[n(\Sigma_k(d)+o(1))]$.

3 Discussion and Related Work

In this section we discuss some relevant related work and also explain the impact of Theorem 1 on some questions that have come up in the literature.

3.1 The k-Colorability Threshold

The problem of determining the chromatic number of random graphs has attracted a great deal of attention since it was first posed by Erdős and Rényi [11] (see [15] for a comprehensive overview). In the case that $p = d/n$ for a fixed real $d > 0$, the problem amounts to calculating the threshold sequence $d_{k-\mathrm{col}}(n)$. The best current bounds are

$$(2k-1)\ln k - 2\ln 2 + \varepsilon_k \le \liminf_{n\to\infty} d_{k-\mathrm{col}}(n) \le \limsup_{n\to\infty} d_{k-\mathrm{col}}(n) \le (2k-1)\ln k - 1 + \delta_k, \quad (3.1)$$

where $\varepsilon_k, \delta_k \to 0$ as $k\to\infty$. The upper bound is by the "first moment" method [7]. The lower bound rests on a "second moment" argument [8], which improves a landmark result of Achlioptas and Naor [3].

While Theorem 1 allows for the possibility that $d_{k,\mathrm{cond}}$ is equal to the k-colorability threshold $d_{k-\mathrm{col}}$ (if it exists), the physics prediction is that these two are different. More specifically, the cavity method yields a prediction as to the precise value of $d_{k-\mathrm{col}}$ in terms of another distributional fixed point problem. An asymptotic expansion in terms of k leads to the conjecture $d_{k-\mathrm{col}} = (2k-1)\ln k - 1 + \eta_k$ with $\eta_k \to 0$ as $k\to\infty$. Thus, the upper bound in (3.1) is conjectured to be asymptotically tight in the limit $k\to\infty$.

The present work builds upon the second moment argument from [8]. Conversely, Theorem 1 yields a small improvement over the lower bound from [8]. Indeed, as we saw above Theorem 1 implies that $\liminf_{n\to\infty} d_{k-\mathrm{col}}(n) \ge d_{k,\mathrm{cond}}$, thereby determining the precise "error term" ε_k in the lower bound (3.1). In fact, $d_{k,\mathrm{cond}}$ is the best-possible lower bound that can be obtained via a certain "natural" type of second moment argument.

3.2 "Quiet Planting?"

The notion that for d close to the (hypothetical) k-colorability threshold $d_{k-\mathrm{col}}$ it seems difficult to find a k-coloring of $G(n, d/n)$ algorithmically could be used to construct a candidate one-way function [2] (see also [14]). This function maps a k-coloring σ to a random graph $G(n, p', \sigma)$ by linking any two vertices v, w with $\sigma(v) \neq \sigma(w)$ with some p' independently. The edge probability p' could be chosen such that the average degree of the resulting graph is close to the k-colorability threshold. The resulting distribution on graphs is the so-called *planted model*.

If the planted distribution is close to $G(n, d/n)$, one might think that the function $\sigma \mapsto G(n, p', \sigma)$ is difficult to invert. Indeed, it should be difficult to find *any* k-coloring of $G(n, p', \sigma)$, not to mention the planted coloring σ. As shown in [2], the planted distribution and $G(n, d/n)$ are interchangeable (in a certain precise sense) iff $\Phi_k(d) = k(1-1/k)^{d/2}$. Hence, $d_{k,\mathrm{cond}}$ marks the point where these two distributions start to differ. In particular, Theorem 1 shows that at the k-colorability threshold, the two distributions are *not* interchangeable.

3.3 Message Passing Algorithms

The cavity method has inspired new "message passing" algorithms by the name of Belief/Survey Propagation Guided Decimation [26]. Experiments on random graph k-coloring instances for small values of k show an excellent performance of these algorithms [4, 30, 22]. However, whether these experimental results are reliable and/or extend to larger k remains shrouded in mystery.

For instance, Belief Propagation Guided Decimation can most easily be described in terms of list colorings. Suppose that G is a given input graph. Initially, the list of colors available to each vertex is the full set $[k]$. The algorithm chooses a color for one vertex at a time as follows. First, it performs a certain fixed point iteration to approximate for each vertex the marginal probability of taking some color i in a randomly chosen proper list coloring of G. Then, a vertex v is chosen, say, uniformly at random and a random color i is chosen from the (supposed) approximation to its marginal distribution. The color list of v is reduced to the singleton $\{i\}$, color i gets removed the lists of all the neighbors of v, and we repeat. The algorithm terminates when either for each vertex a color has been chosen ("success") or the list of some vertex becomes empty ("failure"). Ideally, if at each step the algorithm manages to compute precisely the correct marginal distribution, the result would be a uniformly random k-coloring of the input graph. Of course, generating such a random k-coloring is $\#P$-hard in the worst case, and the crux is that the aforementioned fixed point iteration may or may not produce a good approximation to the actual marginal distribution.

Perhaps the most plausible stab at understanding Belief Propagation Guided Decimation is the non-rigorous contribution [28]. Roughly speaking, the result of the Belief Propagation fixed point iteration after t iterations can be expected to yield a good approximation to the actual marginal distribution iff there is no condensation among the remaining list colorings. If so, one should expect that the algorithm actually finds a k-coloring if condensation does not occur at any step $0 \leq t \leq n$. Thus, we look at a two-dimensional "phase diagram" parametrised by the average degree d and the time t/n. We need to identify the line that marks the (suitably defined) condensation phase transition in this diagram. Theorem 1 deals with the case $t = 0$, and it would be most interesting to see if the present techniques extend to $t \in (0, 1)$. Attempts at (rigorously) analysing message passing algorithms along these lines have been made for random k-SAT, but the results have been far from precise [6, 9].

3.4 The Physics Perspective

In physics terminology the random graph coloring problem is an example of a "diluted mean-field model of a disordered system". The term "mean-field" refers to the fact that there is no underlying lattice geometry, while "diluted" indicates that the average degree in the underlying graph is bounded. Moreover, "disordered systems" reflects that the model involves some degree of randomness (i. e., the random graph). Diluted mean-field models are considered a better approximation to "real" disordered systems (such as glasses) than models where the underlying graph is complete, the Sherrington-Kirkpatrick model [25]. From the viewpoint of physics, the question of whether "disordered systems" exhibit a condensation phase transition can be traced back to Kauzmann's experiments in the 1940s [16]. In models where the underlying graph is complete, physicsts predicted an affirmative answer in the 1980s [17], and this has long been confirmed rigorously [29].

With respect to "diluted" models, Coja-Oghlan and Zdeborova [10] showed that a condensation phase transition *exists* in random r-uniform hypergraph 2-coloring. Furthermore, [10] determines the location of the condensation phase transition up to an error ε_r that tends to zero as the uniformity r of the hypergraph becomes large. By contrast, Theorem 1 is the first result that pins down the *exact* condensation phase transition in a diluted mean-field model.

Technically, we build upon some of the techniques that have been developed to study the "geometry" of the set of k-colorings of the random graph, and add to this machinery. Among the techniques that we harness is the "planting trick" from [2] (which, in a sense, we are going to "put into reverse"), the notion of a core [2, 8, 27], techniques for proving the existence of "frozen variables" [27], and a concentration argument from [10]. Additionally, our proof directly incorporates some of the physics calculations from [22, Appendix C]. That said, the cornerstone of the present work is a novel argument that allows us to connect the distributional fixed point problem from [22] rigorously with the geometry of the set of k-colorings.

4 Proof Outline

From now on we assume that $k \geq k_0$ for some large enough constant k_0.

The proof of Theorem 1 is composed of two parallel threads. The first thread is to identify an "obvious" point where a phase transition occurs or, more specifically, a critical degree $d_{k,\mathrm{crit}}$ where statements (i)-(iii) of the theorem are met. The second thread is to identify the frozen fixed point $\pi^*_{d,k}$ of $\mathcal{F}_{d,k}$ and to interpret it combinatorially. Finally, the two threads intertwine to show that $d_{k,\mathrm{crit}} = d_{k,\mathrm{cond}}$, i. e. that the "obvious" phase transition $d_{k,\mathrm{crit}}$ is indeed the unique zero of equation (2.6). The first thread is an extension of ideas developed in [10] for random hypergraph 2-coloring to the (technically far more involved) random graph coloring problem. The second thread and the intertwining of the two require novel arguments.

4.1 The First Thread

Because the nth root sits inside the expectation, the quantity

$$\Phi_k(d) = \lim_{n\to\infty} \mathbb{E}[Z_k(G(n, d/n))^{1/n}]$$

is difficult to calculate for general values of d. However for $d \in [0, 1)$, $\Phi_k(d)$ is easily understood. In fact, the celebrated result of Erdős and Rényi [11] implies that for $d \in [0, 1)$

the random graph $G(n, d/n)$ is basically a forest. Moreover, the number of k-colorings of a forest with n vertices and m edges is well-known to be $k^n(1-1/k)^m$. Since $G(n, d/n)$ has $m \sim dn/2$ edges w.h.p., we obtain

$$Z_k(G(n,d/n))^{1/n} \sim k(1-1/k)^{d/2} \qquad \text{for } d < 1. \tag{4.1}$$

As $Z_k(G)^{1/n} \leq k$ for any graph on n vertices, (4.1) implies that

$$\Phi_k(d) = \lim_{n\to\infty} \mathbb{E}[Z_k(G(n,d/n))^{1/n}] = k(1-1/k)^{d/2} \qquad \text{for } d < 1. \tag{4.2}$$

Clearly, the function $d \mapsto k(1-1/k)^{d/2}$ is analytic on all of $(0,\infty)$. Therefore, the uniqueness of analytic continuations implies that the least $d > 0$ where the limit $\Phi_k(d)$ either fails to exist or differs from $k(1-1/k)^{d/2}$ is going to be a phase transition. Hence, we let

$$d_{k,\mathrm{crit}} = \sup\left\{d \geq 0 : \text{the limit } \Phi_k(d) \text{ exists and } \Phi_k(d) = k(1-1/k)^{d/2}\right\}. \tag{4.3}$$

▶ **Fact 3.** *We have* $d_{k,\mathrm{crit}} \leq (2k-1)\ln k$.

Thus, $d_{k,\mathrm{crit}}$ is a well-defined finite number, and there occurs a phase transition at $d_{k,\mathrm{crit}}$. Moreover, the following proposition yields a lower bound on $d_{k,\mathrm{crit}}$ and implies that $d_{k,\mathrm{crit}}$ satisfies the first condition in Theorem 1.

▶ **Proposition 4.** *For any* $d > 0$ *we have* $\limsup_{n\to\infty} \mathbb{E}[Z_k(G(n,d/n))^{1/n}] \leq k(1-1/k)^{d/2}$. *Moreover, the number* $d_{k,\mathrm{crit}}$ *satisfies*

$$d_{k,\mathrm{crit}} = \sup\left\{d \geq 0 : \liminf_{n\to\infty} \mathbb{E}[Z_k(G(n,d/n))^{1/n}] \geq k(1-1/k)^{d/2}\right\} \geq (2k-1)\ln k - 2. \tag{4.4}$$

Thus, we know that there *exists* a number $d_{k,\mathrm{crit}}$ that satisfies conditions (i)–(ii) in Theorem 1. Of course, to actually calculate this number we need to unearth its combinatorial "meaning". As we saw in Section 2, if $d_{k,\mathrm{crit}}$ really is the condensation phase transition, then the combinatorial interpretation should be as follows. For $d < d_{k,\mathrm{crit}}$, the size of the cluster that a randomly chosen k-coloring $\boldsymbol{\tau}$ belongs to is smaller than $Z_k(G(n,d/n))$ by an exponential factor $\exp(\Omega(n))$ w.h.p. But as d approaches $d_{k,\mathrm{crit}}$, the gap between the cluster size and $Z_k(G(n,d/n))$ diminishes. Hence, $d_{k,\mathrm{crit}}$ should mark the point where the cluster size has the same order of magnitude as $Z_k(G(n,d/n))$.

But how can we possibly get a handle on the size of the cluster that a randomly chosen k-coloring $\boldsymbol{\tau}$ of $G(n,d/n)$ belongs to? No "constructive" argument (or efficient algorithm) is known for obtaining a single k-coloring of $G(n,d/n)$ for d anywhere close to $d_{k,\mathrm{crit}}$, let alone for sampling one uniformly at random. Nevertheless, as observed in [2], in the case that $\Phi_k(d) = k(1-1/k)^{d/2}$, i. e., for $d < d_{k,\mathrm{crit}}$, it is possible to capture the experiment of first choosing the random graph $G(n,d/n)$ and then sampling a k-coloring $\boldsymbol{\tau}$ uniformly at random by means of a different, much more innocent experiment.

In this latter experiment, we *first* choose a map $\boldsymbol{\sigma} : [n] \to [k]$ uniformly at random. Then, we generate a graph $G(n,p',\boldsymbol{\sigma})$ on $[n]$ by connecting any two vertices $v, w \in [n]$ such that $\boldsymbol{\sigma}(v) \neq \boldsymbol{\sigma}(w)$ with probability p' independently. If $p' = dk/(k-1)$ is chosen so that the expected number of edges is the same as in $G(n,d/n)$, then this so-called *planted model* might be a good approximation to the "difficult" experiment of first choosing $G(n,d/n)$ and then picking a random k-coloring. In particular, we might expect that

$$\mathbb{E}[|\mathcal{C}(G(n,p',\boldsymbol{\sigma}),\boldsymbol{\sigma})|^{1/n}] \sim \mathbb{E}[|\mathcal{C}(G(n,d/n),\boldsymbol{\tau})|^{1/n}],$$

i. e., that the suitably scaled cluster size in the planted model is about the same as the cluster size in $G(n, d/n)$. Hence, $d_{k,\mathrm{crit}}$ should mark the point where $\mathbb{E}[|\mathcal{C}(G(n,p',\boldsymbol{\sigma}),\boldsymbol{\sigma})|^{1/n}]$ equals $k(1-1/k)^{d/2}$. The following Proposition verifies that this is indeed so. Let us write $\boldsymbol{G} = G(n,p',\boldsymbol{\sigma})$ for the sake of brevity.

▶ **Proposition 5.** *Assume that* $(2k-1)\ln k - 2 \le d \le (2k-1)\ln k$ *and set*

$$p' = d'/n \quad \text{with } d' = \frac{dk}{k-1}. \tag{4.5}$$

1. *If*

$$\lim_{\varepsilon\searrow 0}\liminf_{n\to\infty}\mathbb{P}\left[|\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})|^{1/n} \le k(1-1/k)^{d/2}-\varepsilon\right] = 1, \tag{4.6}$$

   then $d \le d_{k,\mathrm{crit}}$.
2. *Conversely, if*

$$\lim_{\varepsilon\searrow 0}\liminf_{n\to\infty}\mathbb{P}\left[|\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})|^{1/n} \ge k(1-1/k)^{d/2}+\varepsilon\right] = 1, \tag{4.7}$$

   then $\limsup_{n\to\infty}\mathbb{E}[Z_k(G(n,d/n))^{1/n}] < k(1-1/k)^{d/2}$. *In particular,* $d \ge d_{k,\mathrm{crit}}$.

4.2 The Second Thread

Our next aim is to "solve" the fixed point problem for $\mathcal{F}_{d,k}$ to an extent that gives the fixed point an explicit combinatorial interpretation. This combinatorial interpretation is in terms of a certain random tree process, associated with a concept of "legal colorings". Specifically, we consider a multi-type Galton-Watson branching process. Its set of types is

$$\mathcal{T} = \{(i,\ell) : i \in [k]\,,\ \ell \subset [k]\,,\ i \in \ell\}\,.$$

The intuition is that i is a "distinguished color" and that ℓ is a set of "available colors". The branching process is further parameterized by a vector $\boldsymbol{q} = (q_1,\ldots,q_k) \in [0,1]^k$ such that $q_1+\cdots+q_k \le 1$. Let $d' = dk/(k-1)$ and

$$q_{i,\ell} = \frac{1}{k}\prod_{j\in\ell\setminus\{i\}}\exp(-q_j d')\cdot\prod_{j\in[k]\setminus\ell}1-\exp(-q_j d') \qquad \text{for } (i,\ell)\in\mathcal{T}.$$

Then

$$\sum_{(i,\ell)\in\mathcal{T}} q_{i,\ell} = 1.$$

Further, for each $(i,\ell)\in\mathcal{T}$ such that $|\ell|>1$ we define $\mathcal{T}_{i,\ell}$ as the set of all $(i',\ell')\in\mathcal{T}$ such that $\ell\cap\ell'\neq\emptyset$ and $|\ell'|>1$. In addition, for $(i,\ell)\in\mathcal{T}$ such that $|\ell|=1$ we set $\mathcal{T}_{i,\ell}=\emptyset$.

The branching process $\mathrm{GW}(d,k,\boldsymbol{q})$ starts with a single individual, whose type $(i,\ell)\in\mathcal{T}$ is chosen from the probability distribution $(q_{i,\ell})_{(i,\ell)\in\mathcal{T}}$. In the course of the process, each individual of type $(i,\ell)\in\mathcal{T}$ spawns a Poisson number $\mathrm{Po}(d'q_{i',\ell'})$ of offspring of type (i',ℓ') for each $(i',\ell')\in\mathcal{T}_{i,\ell}$. In particular, only the initial individual may have a type (i,ℓ) with $|\ell|=1$, in which case it does not have any offspring. Let $1\le\mathcal{N}\le\infty$ be the progeny of the process (i. e., the total number of individuals created).

We are going to view $\mathrm{GW}(d,k,\boldsymbol{q})$ as a distribution over trees endowed with some extra information. Let us define a *decorated graph* as a graph $T=(V,E)$ together with a map $\vartheta: V\to\mathcal{T}$ such that for each edge $e=\{v,w\}\in E$ we have $\vartheta(w)\in\mathcal{T}_{\vartheta(v)}$. Moreover, a *rooted*

decorated graph is a decorated graph (T, ϑ) together with a distinguished vertex v_0, the *root*. Further, an *isomorphism* between two rooted decorated graphs T and T' is an isomorphism of the underlying graphs that preserves the root and the types of the vertices.

Given that $\mathcal{N} < \infty$, the branching process $\mathrm{GW}(d, k, \boldsymbol{q})$ canonically induces a probability distribution over isomorphism classes of rooted decorated trees. Indeed, we obtain a tree whose vertices are all the individuals created in the course of the branching process and where there is an edge between each individual and its offspring. The individual from which the process started is the root. Moreover, by construction each individual v comes with a type $\vartheta(v)$. We denote the (random) isomorphism class of this tree by $\boldsymbol{T}_{d,k,\boldsymbol{q}}$. (It is natural to view the branching process as a probability distribution over *isomorphism classes* as the process does not specify the order in which offspring is created.)

To proceed, we define a *legal coloring* of a decorated graph (G, ϑ) as a map $\tau : V(G) \to [k]$ such that τ is a k-coloring of G and such that for any type $(i, \ell) \in \mathcal{T}$ and for any vertex v with $\vartheta(v) = (i, \ell)$ we have $\tau(v) \in \ell$. Combinatorially, if $\vartheta(v) = (i, \ell)$, then we think of ℓ as a list of colors available to v and of i as a "distinguished color". Let $\mathcal{Z}(G, \vartheta)$ denote the number of legal colorings.

Since $\mathcal{Z}(G, \vartheta)$ is isomorphism-invariant, we obtain the integer-valued random variable $\mathcal{Z}(\boldsymbol{T}_{d,k,\boldsymbol{q}})$. We have $\mathcal{Z}(\boldsymbol{T}_{d,k,\boldsymbol{q}}) \geq 1$ with certainty because a legal coloring τ can be constructed by coloring each vertex with its distinguished color (i. e., setting $\tau(v) = i$ if v has type (i, ℓ)). Hence, $\ln \mathcal{Z}(\boldsymbol{T}_{d,k,\boldsymbol{q}})$ is a well-defined non-negative random variable. Additionally, we write $|\boldsymbol{T}_{d,k,\boldsymbol{q}}|$ for the number of vertices in $\boldsymbol{T}_{d,k,\boldsymbol{q}}$.

Finally, consider a rooted, decorated tree (T, ϑ, v_0) and let $\boldsymbol{\tau}$ be a legal coloring of (T, ϑ, v_0) chosen uniformly at random. Then the color $\boldsymbol{\tau}(v_0)$ of the root is a random variable with values in $[k]$. Let $\mu_{T,\vartheta,v_0} \in \Omega$ denote the distribution of this random variable. Clearly, μ_{T,ϑ,v_0} is invariant under isomorphisms. Consequently, the distribution $\mu_{\boldsymbol{T}_{d,k,\boldsymbol{q}}}$ of the color of the root of a tree in the random isomorphism class $\boldsymbol{T}_{d,k,\boldsymbol{q}}$ is a well-defined Ω-valued random variable. Let $\pi_{d,k,\boldsymbol{q}} \in \mathcal{P}$ denote its distribution. Then we can characterise the frozen fixed point of $\mathcal{F}_{d,k}$ as follows.

▶ **Proposition 6.** *Suppose that* $d \geq (2k-1)\ln k - 2$.

1. *The function*

$$q \in [0,1] \mapsto (1 - \exp(-dq/(k-1)))^{k-1} \tag{4.8}$$

has a unique fixed point q^* *in the interval* $[2/3, 1]$. *Moreover, with*

$$\boldsymbol{q}^* = k^{-1}(q^*, \ldots, q^*) \in [0,1]^k \tag{4.9}$$

the branching process $\mathrm{GW}(d, k, \boldsymbol{q}^*)$ *is sub-critical. Thus,* $\mathbb{P}[\mathcal{N} < \infty] = 1$.
2. *The map* $\mathcal{F}_{d,k}$ *has precisely one frozen fixed point, namely* $\pi_{d,k,\boldsymbol{q}^*}$.
3. *We have* $\phi_{d,k}(\pi_{d,k,\boldsymbol{q}^*}) = \mathbb{E}\left[\frac{\ln \mathcal{Z}(\boldsymbol{T}_{d,k,\boldsymbol{q}^*})}{|\boldsymbol{T}_{d,k,\boldsymbol{q}^*}|}\right]$.
4. *The function* Σ_k *from (2.6) is strictly decreasing and continuous on* $[(2k-1)\ln k - 2, (2k-1)\ln k - 1]$ *and has a unique zero* $d_{k,\mathrm{cond}}$ *in this interval.*

4.3 Tying Up the Threads

To prove that $d_{k,\mathrm{cond}} = d_{k,\mathrm{crit}}$, we establish a connection between the random tree $\boldsymbol{T}_{d,k,\boldsymbol{q}^*}$ and the random graph $\boldsymbol{G}$ with planted coloring $\boldsymbol{\sigma}$. We start by giving a recipe for computing the cluster size $|\mathcal{C}(\boldsymbol{G}, \boldsymbol{\sigma})|$, and then let the random tree process "cook" it.

Computing the cluster size hinges on a close understanding of its combinatorial structure. As hypothesised in physics work [25] and established rigorously in [2, 7, 27], typically many vertices v are "frozen" in $\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})$, i. e., $\tau(v)=\tau'(v)$ for any two colorings $\tau,\tau'\in\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})$. More generally, we consider for each vertex v the set

$$\ell(v)=\{\tau(v):\tau\in\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})\}$$

of colors that v may take in colorings τ that belong to the cluster. Together with the "planted" color $\boldsymbol{\sigma}(v)$, we can thus assign each vertex v a type $\vartheta(v)=(\boldsymbol{\sigma}(v),\ell(v))$. This turns $\boldsymbol{G}$ into a decorated graph $(\boldsymbol{G},\vartheta)$.

By construction, each coloring $\tau\in\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})$ is a legal coloring of the decorated graph $\boldsymbol{G}$. Conversely, it turns out that w.h.p. any legal coloring of $(\boldsymbol{G},\vartheta)$ belongs to the cluster $\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})$. Hence, computing the cluster size $|\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})|$ amounts to calculating the number $\mathcal{Z}(\boldsymbol{G},\vartheta)$ of legal colorings of $\boldsymbol{G},\vartheta$.

This calculation is facilitated by the following observation. Let $\widetilde{\boldsymbol{G}}$ be the graph obtained from $\boldsymbol{G}$ by deleting all edges $e=\{v,w\}$ that join two vertices such that $\ell(v)\cap\ell(w)=\emptyset$. Then any legal coloring τ of $\widetilde{\boldsymbol{G}}$ is a legal coloring of $\boldsymbol{G}$, because $\tau(v)\in\ell(v)$ for any vertex v. Hence, $\mathcal{Z}(\boldsymbol{G},\vartheta)=\mathcal{Z}(\widetilde{\boldsymbol{G}},\vartheta)$.

Thus, we just need to compute $\mathcal{Z}(\widetilde{\boldsymbol{G}},\vartheta)$. This task is much easier than computing $\mathcal{Z}(\boldsymbol{G},\vartheta)$ directly because $\widetilde{\boldsymbol{G}}$ turns out to have *significantly* fewer edges than $\boldsymbol{G}$ w.h.p. More precisely, w.h.p. $\widetilde{\boldsymbol{G}}$ (mostly) consists of connected components that are trees of bounded size. In fact, in a certain sense the distribution of the tree components converges to that of the decorated random tree $\boldsymbol{T}_{d,k,\boldsymbol{q}_*}$. In effect, we obtain

▶ **Proposition 7.** *Suppose that $d\geq(2k-1)\ln k-2$ and let p' be as in (4.5). Let $\boldsymbol{q}^*$ be as in (4.9). Then the sequence $\{\frac{1}{n}\ln|\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})|\}_n$ converges to $\mathbb{E}\left[\frac{\ln\mathcal{Z}(\boldsymbol{T}_{d,k,\boldsymbol{q}^*})}{|\boldsymbol{T}_{d,k,\boldsymbol{q}^*}|}\right]$ in probability.*

The proof of Proposition 7 is the centrepiece of this work. It is based on the precise analysis of a further message-passing algorithm called *Warning Propagation* on the random graph $(\boldsymbol{G},\boldsymbol{\sigma})$ chosen from the planted model. The following section contains an outline of this analysis Combining Propositions 5 and 7, we see that $d_{k,\mathrm{crit}}$ is equal to $d_{k,\mathrm{cond}}$ given by Proposition 6. Theorem 1 then follows from Proposition 4.

5 The Cluster Size

5.1 Warning Propagation

A key step towards the proof of Proposition 7 is to determine the set

$$\ell(v)=\{\tau(v):\tau\in\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})\}$$

of colors that a vertex v may take under a k-coloring in $\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})$. In particular, we called a vertex *frozen* if $\ell(v)=\{\boldsymbol{\sigma}(v)\}$. To establish Proposition 7, we will first show that the sets $\ell(v)$ can be determined by means of a message-passing algorithm called *Warning Propagation* ("WP") [25]. WP has been previously analysed on planted k-SAT instances (where it is similar to Unit Clause Propagation) to show that the algorithm actually finds a solution w.h.p. under certain assumptions [12]. Moreover, the work [27] on frozen variables in k-coloring has a WP flavour. But here we use WP to achieve an even more delicate objective: we aim to figure out the *number* of solutions in the cluster of the planted k-coloring.

More precisely, we will see that WP yields color sets $L(v)$ such that $L(v)=\ell(v)$ for all but $o(n)$ vertices w.h.p. Crucially, by tracing WP we will be able to determine for any given

type (i, ℓ) how many vertices of that type there are. Moreover, we will show that the cluster essentially consists of all k-colorings τ of $\boldsymbol{G}$ such that $\tau(v) \in L(v)$ for all v. In addition, the number of such colorings τ can be calculated by considering a reduced graph $\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma})$. This graphs turns out to be mainly a forest, and finally, informally speaking, w.h.p. the statistics of the trees in this forest coincide with the distribution of the random tree $\boldsymbol{T}_{d,k,\boldsymbol{q}^*}$.

Let us begin by describing Warning Propagation on a general graph G endowed with a k-coloring σ. For each edge $e = \{v, w\}$ of G and any color i we define a sequence $(\mu_{v\to w}(i, t|G, \sigma))_{t\geq 1}$ such that $\mu_{v\to w}(i, t|G, \sigma) \in \{0, 1\}$ for all i, v, w, t. The idea is that $\mu_{v\to w}(i, t|G, \sigma) = 1$ indicates that in the tth step of the process vertex v "warns" vertex w that the other neighbors $u \neq w$ of v force v to take color i. We initialize this process by having each vertex v emit a warning about its original $\sigma(v)$ at $t = 0$, i. e.,

$$\mu_{v\to w}(i, 0|G, \sigma) = \mathbf{1}_{i=\sigma(v)} \tag{5.1}$$

for all edges $\{v, w\}$ and all $i \in [k]$. Letting $\partial v = \partial_G(v)$ denote the neighborhood of v in G, for $t \geq 0$ we let

$$\mu_{v\to w}(i, t+1|G, \sigma) = \prod_{j\in[k]\setminus\{i\}} \max\left\{\mu_{u\to v}(j, t|G, \sigma) : u \in \partial v \setminus \{w\}\right\}. \tag{5.2}$$

That is, v warns w about color i in step $t + 1$ iff at step t it received warnings from its other neighbors u (*not* including w) about all colors $j \neq i$. Further, for a vertex v and $t \geq 0$ we let

$$L(v, t|G, \sigma) = \left\{j \in [k] : \max_{u\in\partial v} \mu_{u\to v}(j, t|G, \sigma) = 0\right\} \quad \text{and} \quad L(v|G, \sigma) = \bigcup_{t=0}^{\infty} L(v, t|G, \sigma).$$

Thus, $L(v, t|G, \sigma)$ is the set of colors that vertex v receives no warnings about at step t. To unclutter the notation, we omit the reference to G, σ where it is apparent from the context.

To understand the semantics of this process, observe that by construction the list $L(v, t|G, \sigma)$ only depends on the vertices at distance at most $t + 1$ from v. Further, if we assume that the tth neighborhood $\partial^t v$ in G is a tree, then $L(v, t|G, \sigma)$ is precisely the set of colors that v may take in k-colorings τ of G such that $\tau(w) = \sigma(w)$ for all vertices w at distance greater than t from v. (This can be verified by a straightforward induction on t.) As we will see, this observation together with the fact that the random graph $\boldsymbol{G}$ contains only few short cycles allows us to show that for most vertices v we have $\ell(v) = L(v|\boldsymbol{G}, \boldsymbol{\sigma})$ w.h.p. In effect, the number of k-colorings τ of $\boldsymbol{G}$ with $\tau(v) \in L(v|\boldsymbol{G}, \boldsymbol{\sigma})$ for all v will emerge to be a very good approximation to the cluster size $\mathcal{C}(\boldsymbol{G}, \boldsymbol{\sigma})$.

Counting these k-colorings τ is made possible by the following observation. For a graph G together with a k-coloring σ, let us denote by $G_{\mathrm{WP}}(t|\sigma)$ the graph obtained from G by removing all edges $\{v, w\}$ such that either $|L(v, t)| < 2$, $|L(w, t)| < 2$ or $L(v, t) \cap L(w, t) = \emptyset$. Furthermore, obtain $G_{\mathrm{WP}}(\sigma)$ from G by removing all edges $\{v, w\}$ such that $L(v) \cap L(w) = \emptyset$. We view $G_{\mathrm{WP}}(t|\sigma)$ and $G_{\mathrm{WP}}(\sigma)$ as decorated graphs in which each vertex v is endowed with the color list $L(v, t)$ and $L(v)$ respectively. As before, we let $\mathcal{Z}$ denote the number of legal colorings of a decorated graph. The key statement in this section is

▶ **Proposition 8.** *W.h.p. we have* $\ln \mathcal{Z}(\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma})) = \ln |\mathcal{C}(\boldsymbol{G}, \boldsymbol{\sigma})| + o(n)$.

We begin by proving that $\mathcal{Z}(\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma}))$ is a lower bound on the cluster size w.h.p. First we are going to argue that w.h.p. in $\boldsymbol{G}$ there are many of frozen vertices, and that thus *all* legal colorings τ of $\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma})$ belong to the cluster $\mathcal{C}(\boldsymbol{G}, \boldsymbol{\sigma})$ w.h.p. To exhibit frozen vertices, we

consider an appropriate notion of a "core". More precisely, assume that σ is a k-coloring of a graph G. We denote by $\mathrm{core}(G,\sigma)$ the largest set V' of vertices with the following property.

$$\text{If } v \in V' \text{ and } j \neq \sigma(v), \text{ then } |V' \cap \sigma^{-1}(j) \cap \partial v| \geq 100. \tag{5.3}$$

In words, any vertex in the core has at least 100 neighbors of any color $j \neq \sigma(v)$ that also belong to the core. The core is well-defined; for if V', V'' are two sets with this property, then so is $V' \cup V''$. The following is immediate from the definition.

▶ **Fact 9.** *Assume that $v \in \mathrm{core}(G,\sigma)$. Then $L(v,t) = \{\sigma(v)\}$ for all t.*

The core has become a standard tool in the theory of random structures in general and in random graph coloring in particular (e. g., [2, 8, 27]). Indeed, standard arguments show that $\boldsymbol{G}$ has a very large core w.h.p.

▶ **Proposition 10** ([8])**.** *W.h.p. we have*

$$|\mathrm{core}(\boldsymbol{G},\boldsymbol{\sigma}) \cap \boldsymbol{\sigma}^{-1}(i)| \geq \frac{n}{k}(1-k^{-2/3}) \quad \text{for all } i \in [k]. \tag{5.4}$$

Moreover, if $v \in \mathrm{core}(\boldsymbol{G},\boldsymbol{\sigma})$, then $\boldsymbol{\sigma}(v) = \tau(v)$ for all $\tau \in \mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})$.

▶ **Corollary 11.** *W.h.p. we have $|\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})| \geq \mathcal{Z}(\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma}))$.*

While $\mathcal{Z}(\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma}))$ provides a lower bound on the cluster size, the two numbers do not generally coincide. This is because for a few vertices v, the set $L(v)$ produced by WP may be a proper subset of $\ell(v)$. (Bipartite sub-structures known as "Kempe chains" are for instance responsable for this, cf. [27].) The origin of this problem is that we launched WP from the initialization (5.1), which is the obvious choice but may be too restrictive. Thus, to obtain an upper bound on the cluster size we will start WP from a different initialization. Ideally, this starting point should be such that only vertices that are frozen emit warnings. By Proposition 10, the vertices in the core meet this condition w.h.p. Thus, we are going to compare the above instalment of Warning Propagation with the result of starting WP from an initialization where only the vertices in the core send out warnings.

Thus, given a graph G together with a k-coloring σ we let

$$\begin{aligned}\mu'_{v\to w}(i,0|G,\sigma) &= \mathbf{1}_{i=\sigma(v)} \cdot \mathbf{1}_{v\in\mathrm{core}(G,\sigma)},\\ \mu'_{v\to w}(i,t+1|G,\sigma) &= \prod_{j\in[k]\setminus\{i\}} \max\{\mu'_{u\to v}(j,t|G,\sigma) : u \in \partial v \setminus \{w\}\}\end{aligned}$$

for all edges $\{v,w\}$ of G, all $i \in [k]$ and all $t \geq 0$. Furthermore, let

$$L'(v,t|G,\sigma) = \left\{j \in [k] : \max_{u\in\partial(v)} \mu'_{u\to v}(j,t|G,\sigma) = 0\right\} \quad \text{and} \quad L'(v|G,\sigma) = \bigcap_{t=0}^{\infty} L'(v,t|G,\sigma).$$

As before, we drop G,σ from the notation where possible.

Similarly as before, we can use the lists $L'(v,t)$ to construct a decorated reduced graph denoted by $G'_{\mathrm{WP}}(t|\sigma)$ and $G'_{\mathrm{WP}}(\sigma)$. Proceeding much as above, we obtain

▶ **Lemma 12.** *W.h.p. we have $|\mathcal{C}(\boldsymbol{G},\boldsymbol{\sigma})| \leq \mathcal{Z}(\boldsymbol{G}'_{\mathrm{WP}}(\boldsymbol{\sigma}))$.*

Combining Corollary 11 and Lemma 12, we see that $\mathcal{Z}(\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma})) \leq |\mathcal{C}(\boldsymbol{G},\sigma)| \leq \mathcal{Z}(\boldsymbol{G}'_{\mathrm{WP}}(\boldsymbol{\sigma}))$ w.h.p. To complete the proof of Proposition 8, we are going to argue that $\ln \mathcal{Z}(\boldsymbol{G}'_{\mathrm{WP}}(\boldsymbol{\sigma})) = \ln \mathcal{Z}(\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma})) + o(n)$ w.h.p.

To this end, we need one more general construction. Let G be a graph and let σ be a k-coloring of G. Let $t \geq 0$ be an integer. For each vertex v of G we define a rooted, decorated graph $T(v,t|G,\sigma)$ as follows.

- The graph underlying $T(v,t|G,\sigma)$ is the connected component of v in $G_{\mathrm{WP}}(t|\sigma)$.
- The root of $T(v,t|G,\sigma)$ is v.
- The type of each vertex w of $T(v,t|G,\sigma)$ is $(\sigma(w), L(w,t|G,\sigma))$.

Analogously we obtain a rooted, decorated graph $T(v|G,\sigma)$ from $G_{\mathrm{WP}}(\sigma)$, $T'(v,t|G,\sigma)$ from $G'_{\mathrm{WP}}(t|\sigma)$ and $T'(v|G,\sigma)$ from $G'_{\mathrm{WP}}(\sigma)$. By carefully coupling our two versions of WP, we obtain

▶ **Lemma 13.** *W.h.p.* $\boldsymbol{G},\boldsymbol{\sigma}$ *is such that* $T(v|\boldsymbol{G},\boldsymbol{\sigma}) = T'(v|\boldsymbol{G},\boldsymbol{\sigma})$ *for all but* $o(n)$ *vertices* v.

5.2 Counting Legal Colorings

Proposition 8 reduces the proof of Proposition 7 to the problem of counting the legal colorings of the reduced graph $\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma})$. For a rooted, decorated tree T let H_T be the number of vertices v in $\boldsymbol{G}_{\mathrm{WP}}(\boldsymbol{\sigma})$ such that $T(v|\boldsymbol{G},\boldsymbol{\sigma}) \cong T$. Let us write $\boldsymbol{T} = \boldsymbol{T}_{d,k,\boldsymbol{q}^*}$ for the sake of brevity. Recall that $\boldsymbol{T}$ is an isomorphism class of rooted, decorated trees; thus, it makes sense to write $T \in \boldsymbol{T}$. To complete the proof of Proposition 7 we need to show the following.

▶ **Proposition 14.** *For any* T *the sequence* $(\frac{1}{n}H_T)_{n\geq 1}$ *converges to* $\mathbb{P}[T \in \boldsymbol{T}]$ *in probability.*

This can be shown by proving that the number q^* from Proposition 6 provides a good approximation to the number of vertices v such that $L(v|\boldsymbol{G},\boldsymbol{\sigma}) = \{i\}$ for any i. As a next step, it can be argued that WP "converges quickly". More specifically, for most vertices v the component $T(v|\boldsymbol{G},\boldsymbol{\sigma})$ is already completely determined after just a bounded number t of iterations of WP. This reduces the proof of Proposition 14 to the problem of studying the statistics of the trees $T(v,t|\boldsymbol{G},\boldsymbol{\sigma})$ with $t \geq 0$ (large but) fixed as $n \to \infty$. This problem is *much* simpler than the original one, because we only need to iterate WP for t rounds. Finally, Proposition 7 follows from Propositions 8 and 14.

Acknowledgment. We thank Guilhem Semerjian for helpful discussions and explanations regarding the articles [20, 22] and Nick Wormald for pointing us to [24, Theorem 3.8].

References

1 D. Achlioptas, E. Friedgut: A sharp threshold for k-colorability. Random Struct. Algorithms **14** (1999) 63–70.
2 D. Achlioptas, A. Coja-Oghlan: Algorithmic barriers from phase transitions. Proc. 49th FOCS (2008) 793–802.
3 D. Achlioptas, A. Naor: The two possible values of the chromatic number of a random graph. Annals of Mathematics **162** (2005) 1333–1349.
4 A. Braunstein, R. Mulet, A. Pagnani, M. Weigt, R. Zecchina: Polynomial iterative algorithms for coloring and analyzing random graphs. Phys. Rev. E **68** (2003) 036702.
5 P. Cheeseman, B. Kanefsky, W. Taylor: Where the *really* hard problems are. Proc. IJCAI (1991) 331–337.
6 A. Coja-Oghlan: On belief propagation guided decimation for random k-SAT. Proc. 22nd SODA (2011) 957–966.
7 A. Coja-Oghlan: Upper-bounding the k-colorability threshold by counting covers. Electronic Journal of Combinatorics **20** (2013) P32.
8 A. Coja-Oghlan, Dan Vilenchik: Chasing the k-colorability threshold. Proc. 54th FOCS (2013) 380–389. A full version is available as arXiv:1304.1063.
9 A. Coja-Oghlan, A. Y. Pachon-Pinzon: The decimation process in random k-SAT. SIAM Journal on Discrete Mathematics **26** (2012) 1471–1509.

10 A. Coja-Oghlan, L. Zdeborová: The condensation transition in random hypergraph 2-coloring. Proc. 23rd SODA (2012) 241–250.

11 P. Erdős, A. Rényi: On the evolution of random graphs. Magayar Tud. Akad. Mat. Kutato Int. Kozl. **5** (1960) 17–61.

12 U. Feige, E. Mossel, D. Vilenchik: Complete convergence of message passing algorithms for some satisfiability problems. Theory of Computing **9** (2013) 617–651.

13 E. Friedgut: Sharp thresholds of graph properties, and the k-SAT problem. J. AMS **12** (1999) 1017–1054.

14 O. Goldreich: Candidate one-way functions based on expander graphs. Electronic Colloquium on Computational Complexity (ECCC) **7** (2000).

15 S. Janson, T. Łuczak, A. Ruciński: Random Graphs, Wiley 2000.

16 W. Kauzmann: The nature of the glassy state and the behavior of liquids at low temperatures. Chem. Rev. **43** (1948) 219–256.

17 T. R. Kirkpatrick, D. Thirumalai: p-spin-interaction spin-glass models: Connections with the structural glass problem. Phys. Rev. B **36** (1987) 5388.

18 F. Krzakala, J. Kurchan: A Landscape Analysis of Constraint Satisfaction Problems. Phys. Rev. E **76** (2007) 02112.

19 F. Krzakala, M. Mézard, F. Sausset, Y. F. Sun, L. Zdeborová: Statistical physics-based reconstruction in compressed sensing. Phys. Rev. X **2** (2012), 021005.

20 F. Krzakala, A. Montanari, F. Ricci-Tersenghi, G. Semerjian, L. Zdeborova: Gibbs states and the set of solutions of random constraint satisfaction problems. Proc. National Academy of Sciences **104** (2007) 10318–10323.

21 F. Krzakala, L. Zdeborová: Generalization of the cavity method for adiabatic evolution of Gibbs states. Phys. Rev. B **81** (2010) 224205.

22 F. Krzakala, L. Zdeborová: Phase transition in the coloring of random graphs. Phys. Rev. E **76** (2007) 031131.

23 H. Levesque, D. Mitchell, B. Selman: Hard and easy distribution of SAT problems. Proc. 10th AAAI (1992) 459–465.

24 C. McDiarmid: Concentration. In Habib et al. (eds): Probabilistic methods for algorithmic discrete mathematics. Springer (1998) 195–248.

25 M. Mézard, A. Montanari: Information, physics and computation. Oxford University Press 2009.

26 M. Mézard, G. Parisi, R. Zecchina: Analytic and algorithmic solution of random satisfiability problems. Science **297** (2002) 812–815.

27 M. Molloy: The freezing threshold for k-colourings of a random graph. Proc. 43rd STOC (2012) 921–930.

28 F. Ricci-Tersenghi, G. Semerjian: On the cavity method for decimated random constraint satisfaction problems and the analysis of belief propagation guided decimation algorithms. J. Stat. Mech. (2009) P09001.

29 M. Talagrand: Spin glasses, a Challenge for Mathematicians. Springer 2003.

30 L. Zdeborová: Statistical Physics of Hard Optimization Problems. Acta Physica Slovaca **59** (2009) 169–303.

The Information Complexity of Hamming Distance

Eric Blais[1], Joshua Brody[2], and Badih Ghazi[1]

1 **MIT, Cambridge, MA, USA**
{eblais|badih}@mit.edu
2 **Swarthmore College, Swarthmore, PA, USA**
joshua.e.brody@gmail.com

Abstract

The Hamming distance function $\textsc{Ham}_{n,d}$ returns 1 on all pairs of inputs x and y that differ in at most d coordinates and returns 0 otherwise. We initiate the study of the information complexity of the Hamming distance function.

We give a new optimal lower bound for the information complexity of the $\textsc{Ham}_{n,d}$ function in the small-error regime where the protocol is required to err with probability at most $\epsilon < d/n$. We also give a new conditional lower bound for the information complexity of $\textsc{Ham}_{n,d}$ that is optimal in all regimes. These results imply the first new lower bounds on the communication complexity of the Hamming distance function for the shared randomness two-way communication model since Pang and El-Gamal (1986). These results also imply new lower bounds in the areas of property testing and parity decision tree complexity.

1998 ACM Subject Classification F.1.2 Modes of Computation

Keywords and phrases Hamming distance, communication complexity, information complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.465

1 Introduction

The Hamming distance function $\textsc{Ham}_{n,d} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ returns 1 on all pairs of inputs $x, y \in \{0,1\}^n$ that differ in at most d coordinates and returns 0 otherwise. This function is one of the fundamental objects of study in communication complexity. In this setting, Alice receives $x \in \{0,1\}^n$, Bob receives $y \in \{0,1\}^n$, and their goal is to compute the value of $\textsc{Ham}_{n,d}(x,y)$ while exchanging as few bits as possible.

The communication complexity of the Hamming distance function has been studied in various communication models [25, 18, 26, 11, 13], leading to tight bounds on the communication complexity of $\textsc{Ham}_{n,d}$ in many settings. One notable exception to this state of affairs is in the shared randomness two-way communication model in which Alice and Bob share a common source of randomness, they can both send messages to each other, and they are required to output the correct value of $\textsc{Ham}_{n,d}(x,y)$ with probability at least $1-\epsilon$ for each pair of inputs x, y. This can be done with a protocol that uses $O(\min\{n, d\log\frac{d}{\epsilon}\})$ bits of communication [13]. Furthermore, this protocol is quite simple: Alice and Bob simply take a random hash of their strings of length $O(\frac{d^2}{\epsilon})$ and determine if the Hamming distance of these hashes is at most d or not.

Pang and El-Gamal [18] showed that the hashing strategy is optimal when $d = cn$ for some constant $0 < c < 1$ and $0 < \epsilon < \frac{1}{2}$ is constant. With a simple padding argument, their result gives a general lower bound of $\Omega(\min\{d, n-d\})$ bits on the communication complexity of $\textsc{Ham}_{n,d}$.[1] Recently, there has been much interest in the Gap-Hamming Distance variant

[1] The same bound can also be obtained via a simple reduction from a promise version of the Set Disjointness

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 465–489

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

$\mathrm{GHD}_{n,d}$ of the Hamming distance function, where the inputs x and y are promised to be at Hamming distance at most $d - \sqrt{d}$ or at least $d + \sqrt{d}$ of each other. This line of work culminated in the recent proof that the $\Omega(\min\{d, n-d\})$ lower bound also holds for the $\mathrm{GHD}_{n,d}$ function [7, 22, 21]. Since Pang and El-Gamal's result, however, there has been no further progress on lower bounds for the communication complexity of the $\textsc{Ham}_{n,d}$ function and closing the gap between this lower bound and the upper bound of the simple hashing protocol remains an open problem.

In this work, we give new lower bounds on the communication complexity of the Hamming distance function by establishing new bounds on its information complexity. Informally, the information complexity of a function f is the amount of information that Alice and Bob must learn about each other's inputs when executing any protocol that computes f. The idea of using information complexity to lower bound the communication complexity of a function goes back to [8] and has since led to a number of exciting developments in communication complexity and beyond ([1, 2, 5, 24] to name just a few).

Let $\mathrm{IC}_\mu(f, \epsilon)$ denote the minimum amount of information that Alice and Bob can reveal to each other about their inputs while computing the function f with probability $1 - \epsilon$ (on every input pair), when their inputs are drawn from the distribution μ. The information complexity of f, denoted $\mathrm{IC}(f, \epsilon)$, is the maximum value of $\mathrm{IC}_\mu(f, \epsilon)$ over all distributions μ on the domain of f. A natural extension of the simple hashing protocol that gives the best-known upper bound on the communication complexity of $\textsc{Ham}_{n,d}$ also yields the best-known upper bound on its information complexity.

▶ **Proposition 1.1.** *For every* $0 < d < n - 1$ *and every* $0 \leq \epsilon < 1/2$,

$$\mathrm{IC}(\textsc{Ham}_{n,d}, \epsilon) \leq O(\min\{\log \tbinom{n}{d}, d \log \tfrac{d}{\epsilon}\}).$$

This bound on the information complexity of $\textsc{Ham}_{n,d}$ matches the communication complexity bound of the function when ϵ is a constant, but is exponentially smaller (in n) when d is small and ϵ tends to (or equals) 0.

By a reduction from a promise version of the Set Disjointness function and the known lower bound on the information complexity of that function [1], the information complexity of the Hamming distance problem is bounded below by

$$\mathrm{IC}(\textsc{Ham}_{n,d}, \epsilon) \geq \Omega(\min\{d, n-d\}) \tag{1}$$

for every $0 \leq \epsilon < \frac{1}{2}$. (In fact, Kerenidis et al. [15] have shown that the same lower bound also holds for the information complexity of the Gap-Hamming Distance function.) This result shows that the bound in Proposition 1.1 is optimal in the large distance regime, when $d = cn$ for some constant $0 < c < 1$.

The bound in Proposition 1.1 is also optimal when d and ϵ are both constants. In this case, the information complexity of $\textsc{Ham}_{n,d}$ is constant. There are two regimes, however, where the information complexity of the Hamming distance function is not yet well understood: the small-error regime where $\epsilon = o(1)$, and the medium-distance regime where $\omega(1) \leq d \leq o(n)$. In this paper, we introduce new lower bounds on the information complexity of $\textsc{Ham}_{n,d}$ for both of these regimes.

function. The optimal lower bound for the communication complexity of this function, however, was obtained later [14].

1.1 Our Results

1.1.1 Lower Bound for the Small-error Regime

Our first goal is to strengthen the lower bound on the information complexity of $\textsc{Ham}_{n,d}$ in the small-error regimes where $\epsilon = o(1)$ and where $\epsilon = 0$. It is reasonable to expect that for every value $0 \leq d \leq n-1$, the information complexity of every $\textsc{Ham}_{n,d}$ function should depend on either n or ϵ in these regimes. Surprisingly, Braverman [5] showed that this is not the case when $d = 0$. The $\textsc{Ham}_{n,0}$ function corresponds to the Equality function, and Braverman showed that for every $\epsilon \geq 0$, $\mathrm{IC}(\textsc{Equality}, \epsilon) = O(1)$ is bounded above by an absolute constant.

We show that the Equality function is in a sense a pathological example: it is the only Hamming distance function whose information complexity is independent of both n and ϵ.

▶ **Theorem 1.2.** *For every $1 \leq d < n-1$ and every $0 \leq \epsilon < 1/2$,*

$$\mathrm{IC}(\textsc{Ham}_{n,d}, \epsilon) = \Omega(\min\{\log \tbinom{n}{d}, d \log(1/\epsilon)\}).$$

The bound in the theorem matches that of Proposition 1.1 whenever $\epsilon < 1/n$. This shows that the lower bound is optimal in this regime and, notably, that the simple hashing protocol for $\textsc{Ham}_{n,d}$ is optimal among all protocols with low error.

There are two main components in the proof of Theorem 1.2. The first is a lower bound on the $\textsc{Ham}_{n,1\text{vs}.3}$, the promise version of the $\textsc{Ham}_{n,1}$ function where the protocol receives the additional guarantee that the two inputs x and y have Hamming distance exactly 1 or 3. Let μ be the uniform distribution over pairs (x, y) at Hamming distance 1 of each other. We show that every ϵ-error protocol for $\textsc{Ham}_{n,1\text{vs}.3}$ has large information cost over μ.

▶ **Lemma 1.3.** *Fix $\epsilon \geq 0$ and let μ be the uniform distribution over the pairs $(x, y) \sim \{0,1\}^n \times \{0,1\}^n$ at Hamming distance 1 of each other. Then*

$$\mathrm{IC}(\textsc{Ham}_{n,1vs.3}, \epsilon) \geq \mathrm{IC}_\mu(\textsc{Ham}_{n,1vs.3}, \epsilon) = \Omega(\min\{\log n, \log 1/\epsilon\}).$$

The second main component in the proof of Theorem 1.2 is a direct sum theorem (implicitly) due to Bar-Yossef et al. [1].[2] Roughly speaking, this direct sum theorem shows that under appropriate conditions, the information cost of any protocol that computes the AND of k copies of a function f is at least k times the information complexity of f. By observing that every protocol for the $\textsc{Ham}_{n,d}$ function also is a valid protocol for the AND of d copies of $\textsc{Ham}_{n/d,1\text{vs}.3}$, we are able to combine the direct sum theorem and Lemma 1.3 to complete the proof of Theorem 1.2.

1.1.2 Conditional Lower Bound

Theorem 1.2 establishes the optimality of the information complexity bound of Proposition 1.1 in every setting except the medium-distance regime, where $\omega(1) \leq d \leq o(n)$ and ϵ is (somewhat) large. We conjecture that the upper bound is optimal in this setting as well.

▶ **Conjecture 1.4.** *For every $1 \leq d < n-1$ and every $0 \leq \epsilon < 1/2$,*

$$\mathrm{IC}(\textsc{Ham}_{n,d}, \epsilon) = \Omega(\min\{\log \tbinom{n}{d}, d \log(d/\epsilon)\}).$$

[2] The direct sum theorem in [1] is stated for a different notion of information complexity but the proof of this theorem can be extended to yield a direct sum theorem for our setting as well. See Section 3 for the details.

A proof of the conjecture would have a number of interesting consequences. In particular, as we describe in more detail in Section 1.2.1 below, it would yield tight bounds on the communication complexity of $\textsc{Ham}_{n,d}$, on the query complexity of fundamental problems in property testing, and on the parity decision tree complexity of a natural Hamming weight function. A proof of the conjecture would also show that the simple hashing protocol is optimal and, in particular, since that protocol always accepts inputs at Hamming distance at most d from each other, it would confirm that two-sided error does not reduce the information or communication complexity of the Hamming distance function.

Finally, a proof of the conjecture would establish a notable separation between the communication complexity of Hamming distance and set disjointness. Let $\textsc{Disj}_n$ denote the function that returns 1 on the inputs $x, y \in \{0,1\}^n$ iff for every coordinate $i \in [n]$, $x_i = 0$ or $y_i = 0$. Let $\textsc{Disj}_{n,k}$ denote the variant on this problem where Alice and Bob's inputs are promised to have Hamming weight k. As mentioned briefly earlier, it is possible to get lower bounds on the communication complexity of $\textsc{Ham}_{n,d}$ with a reduction from $\textsc{Disj}_{n,(d+1)/2}$. When $d = cn$, and $0 < c < 1$ is a constant, this reduction is tight since both functions have communication complexity $\Theta(n)$ in this setting. However, Håstad and Wigderson [12] (see also [20]) showed that the communication complexity of $\textsc{Disj}_{n,k}$ is $O(k)$, so a proof of Conjecture 1.4 would show that the communication complexity of $\textsc{Ham}_{n,d}$ is asymptotically larger than that of $\textsc{Disj}_{n,(d+1)/2}$ when $d = o(n)$.

We give a conditional proof of Conjecture 1.4. To describe the result, we need to introduce a few notions related to parallel repetition. For a function $f : \{0,1\}^n \to \{0,1\}$ and $k \geq 2$, let $f^k : \{0,1\}^{nk} \to \{0,1\}^k$ denote the function that returns the value of f on k disjoint inputs. A protocol computes f^k with error ϵ if it computes the value of f on *all* k of the disjoint inputs with probability at least $1 - \epsilon$.

▶ **Definition 1.5.** A function $f : \mathcal{X}^n \times \mathcal{Y}^n \to \{0,1\}$ is *majority-hard* for the distribution μ on $\mathcal{X} \times \mathcal{Y}$ and for $\epsilon \geq 0$ if there exists a constant $c > 0$ such that for any $k \geq 2$,

$$\mathrm{IC}_{\mu^k}(\mathrm{Maj}_k \circ f, \epsilon) = \Theta(\mathrm{IC}_{\mu^{\lfloor ck \rfloor}}(f^{\lfloor ck \rfloor}, \epsilon)).$$

The upper bound in the definition trivially holds: a protocol for $\mathrm{Maj}_k \circ f$ can first determine the value of the k instances of f in parallel so $\mathrm{IC}_{\mu^k}(\mathrm{Maj}_k \circ f, \epsilon) \leq \mathrm{IC}_{\mu^k}(f^k, \epsilon)$. We believe that the reverse inequality holds for the $\textsc{Ham}_{n,1}$ function. In fact, we do not know of any distribution μ and any function f that is balanced on μ which is not majority-hard for μ. (Determining whether every such function is indeed majority-hard appears to be an interesting question in its own right; see [23] and [17] for related results.)

Let μ_1 and μ_3 be the uniform distributions over the pairs $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ at Hamming distance 1 and 3 of each other, respectively. Let $\mu = \frac{1}{2}\mu_1 + \frac{1}{2}\mu_3$. We give a conditional proof of Conjecture 1.4 assuming that $\textsc{Ham}_{n,1}$ is a majority-hard function on μ.

▶ **Theorem 1.6.** *If* $\textsc{Ham}_{n,1}$ *is majority-hard over the distribution* μ *described above, then for every* $1 \leq d < n - 1$ *and every* $0 \leq \epsilon < 1/2$*,*

$$\mathrm{IC}(\textsc{Ham}_{n,d}, \epsilon) = \Theta(\min\{\log \tbinom{n}{d}, d \log(d/\epsilon)\}).$$

The proof of Theorem 1.6 follows the same overall structure as the proof of Theorem 1.2: we first establish a lower bound on the information complexity of $\textsc{Ham}_{n,1}$ and then use a direct sum theorem to derive the general lower bound from this result. Both of these components of the proof, however, must be significantly extended to yield the stronger lower bound.

In order to prove Theorem 1.6, we need to extend the result from Lemma 1.3 in two ways. First, we need extend the lower bound on the information complexity to apply to protocols in the average error model. In this model, a protocol has error ϵ under μ if the expected error probability on inputs drawn from μ. (By contrast, until now we have only considered protocols that must err with probability at most ϵ on every possible inputs; even those outside the support of μ.) Second, we need a lower bound that also applies to protocols that are allowed to abort with a constant probability δ. We denote the information complexity of the function f over the distribution μ in the ϵ-average-error δ-average-abortion-probability model by $\mathrm{IC}^{\mathrm{avg}}_{\mu}(f, \epsilon, \delta)$.

▶ **Lemma 1.7.** *Fix $0 \leq \epsilon < \frac{1}{2}$ and $0 \leq \delta < 1$. Let μ be the distribution described above. Then*

$$\mathrm{IC}^{\mathrm{avg}}_{\mu}(\textsc{Ham}_{n,1vs.3}, \epsilon, \delta) = \Omega(\min\{\log n, \log 1/\epsilon\}).$$

One significant aspect of the bound in Lemma 1.7 worth emphasizing is that the information complexity is *independent* of the abortion probability δ.

The second main component of the proof of Theorem 1.6 is another direct sum theorem. In this proof, we use a slightly different decomposition of $\textsc{Ham}_{n,d}$: instead of relating it to the composed function $\mathrm{AND}_d \circ \textsc{Ham}_{n/d,1\mathrm{vs}.3}$, we now use the fact that a protocol for $\textsc{Ham}_{n,d}$ also is a valid protocol for $\mathrm{Maj}_{d/2} \circ \textsc{Ham}_{2n/d,1\mathrm{vs}.3}$. If $\textsc{Ham}_{n,1}$ is majority-hard over the distribution μ, this decomposition shows that any protocol for $\textsc{Ham}_{n,d}$ has information complexity at least $\mathrm{IC}_{\mu^{d'}}(\textsc{Ham}^{d'}_{n,1\mathrm{vs}.3}, \epsilon, \delta)$ for some $d' = \Omega(d)$. We can then apply a recent strong direct sum theorem of Molinaro, Woodruff, and Yaroslavtsev [16] to obtain the desired result.

1.2 Extensions and Applications

1.2.1 Lower Bounds in Other Settings

The lower bounds on the information complexity of $\textsc{Ham}_{n,d}$ in Theorems 1.2 and 1.6 immediately imply corresponding lower bounds on the communication complexity of the same function.

▶ **Corollary 1.8.** *Fix $1 \leq d < n-1$ and $0 \leq \epsilon < \frac{1}{2}$. Then $R^{\mathrm{pub}}(\textsc{Ham}_{n,d}, \epsilon) = \Omega(\min\{\log \binom{n}{d}, d \log \frac{1}{\epsilon}\})$. Furthermore, if $\textsc{Ham}_{n,1}$ is majority-hard, then $R^{\mathrm{pub}}(\textsc{Ham}_{n,d}, \epsilon) = \Theta(\min\{\log \binom{n}{d}, d \log \frac{d}{\epsilon}\})$.*

In turn, the lower bounds on the communication complexity of $\textsc{Ham}_{n,d}$ imply new lower bounds on the query complexity of a number of different property testing problems via the connection introduced in [4].

▶ **Corollary 1.9.** *Fix $k \leq \frac{n}{2}$. At least $\Omega(\min\{k \log n, k \log \frac{1}{\delta}\})$ queries are required to test k-linearity and k-juntas with error δ. Furthermore, if $\textsc{Ham}_{n,1}$ is majority-hard, then $\Theta(k \log k)$ queries are required to test k-linearity and k-juntas with constant error.*

The best current lower bound on the query complexity for testing each property in Corollary 1.9 is $\Omega(k)$, a result that was obtained via a reduction from the Set Disjointness function [4]. Corollary 1.9 shows that replacing this reduction with one from the Hamming distance function yields stronger lower bounds.

Theorems 1.2 and 1.6 also give new lower bounds on the decision tree complexity of boolean functions. A *parity decision tree* is a tree where every internal node of the tree branches according to the parity of a specified subset of the bits of the input $x \in \{0,1\}^n$ and

every leaf is labelled with 0 or 1. The randomized ϵ-error parity decision tree complexity of a function $f : \{0,1\}^n \to \{0,1\}$, denoted $R^{\oplus}_{\epsilon}(f)$, is the minimum depth d such that there exists a distribution D over parity decision trees of depth d where for every $x \in \{0,1\}^n$, the path defined by x on a tree drawn from D leads to a leaf labelled by $f(x)$ with probability at least $1-\epsilon$. For $0 \leq d \leq n$, let $\textsc{Weight}_{n,d} : \{0,1\}^n \to \{0,1\}$ be the function that returns 1 iff the input x has Hamming weight at most d.

▶ **Corollary 1.10.** *Fix $0 < d < n-1$ and $0 \leq \epsilon < \frac{1}{2}$. Then $R^{\oplus}_{\epsilon}(\textsc{Weight}_{n,d}) = \Omega(\min\{\log \binom{n}{d}, d \log \frac{1}{\epsilon}\})$. Furthermore, if $\textsc{Ham}_{n,1}$ is majority-hard, then $R^{\oplus}_{\epsilon}(\textsc{Weight}_{n,d}) = \Theta(\min\{\log \binom{n}{d}, d \log \frac{d}{\epsilon}\})$.*

1.2.2 Symmetric XOR Functions

The Hamming distance functions $\textsc{Ham}_{n,d}$ are contained within a larger class of functions called *symmetric XOR functions*. The function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ is a symmetric XOR function if it can be expressed as $f = h \circ \oplus_n$, where $\oplus_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is the entrywise XOR function and $h : \{0,1\}^n \to \{0,1\}$ is a symmetric boolean function.

The *skip complexity* of a symmetric XOR function $f = h \circ \oplus_n$ is defined as $\Gamma_{+2}(f) = \max\{0 \leq d < \frac{n}{2} : h(d) \neq h(d+2) \vee h(n-d) \neq h(n-d-2)\}$. This complexity measure is closely related to the Paturi complexity of symmetric functions [19]. The proof of Theorem 1.2 can be generalized to give a lower bound on the information complexity of every symmetric XOR function in terms of its skip complexity.

▶ **Theorem 1.11.** *Fix $\epsilon \geq 0$. For every symmetric XOR function $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$,*

$$\mathrm{IC}(f, \epsilon) \geq \Omega(\Gamma_{+2}(f) \cdot \min\{\log n, \log 1/\epsilon\}).$$

The only symmetric XOR functions with skip complexity $\Gamma_{+2}(f) = 0$ are the affine combinations of the Equality and Parity functions. Each of these functions has information complexity $O(1)$, so Theorem 1.11 yields a complete characterization of the set of functions that have constant information complexity when $\epsilon = 0$.

1.2.3 Direct Sum Violations

In 1995, Feder et al. [10] showed that the Equality function violates the direct-sum theorem in the randomized communication complexity model when $\epsilon = o(1)$. Braverman [5] noted that an alternative proof of this fact follows from the fact that the information complexity of the Equality function satisfies $\mathrm{IC}(\textsc{Equality}, \epsilon) = O(1)$.

The tight characterization of the information complexity of $\textsc{Ham}_{n,1}$ obtained by the bounds in Proposition 1.1 and Lemma 1.3 shows that $\textsc{Ham}_{n,1}$ satisfies the direct-sum theorem for randomized communication complexity when $n = \mathrm{poly}(1/\epsilon)$ and violates it otherwise (i.e., when $\log n = o(\log 1/\epsilon)$. This result can be seen as further evidence of the qualitative difference between the complexity of the Equality function and that of the "almost-equality" function $\textsc{Ham}_{n,1}$. See Section 7 for the details.

1.2.4 Composition of the $\textsc{Ham}_{n,1}$ Function

One important difference between the proof of Theorem 1.2 and that of Theorem 1.6 is that whereas the former is obtained by analyzing the composed function $\mathrm{AND}_d \circ \textsc{Ham}_{n,1\mathrm{vs}.3}$, the latter is obtained by analyzing $\mathrm{Maj}_{d/2} \circ \textsc{Ham}_{n,1\mathrm{vs}.3}$. It is natural to ask whether this

switch is necessary—whether the stronger lower bound of Theorem 1.6 could be obtained by considering the composed function $\mathrm{AND}_d \circ \mathrm{HAM}_{n,1\mathrm{vs}.3}$.

The same question can be rephrased to ask whether the bound in Theorem 1.2 is optimal for the function $\mathrm{AND}_d \circ \mathrm{HAM}_{n,1\mathrm{vs}.3}$. We show that it is. Furthermore, we show that a similar upper bound also applies to the function $\mathrm{OR}_k \circ \mathrm{HAM}_{n,1}$, so that in order to obtain the lower bound in Theorem 1.6 via a reduction approach, we must consider another composition function. See Section 8 for the details.

2 Information Complexity Preliminaries

We use standard information-theoretic notation and the following basic facts about entropy and mutual information. See [9] for the basic definitions and the proofs of the following facts.

▶ **Fact 2.1.** *If X can be described with k bits given Y, then $H(X|Y) \leq k$.*

▶ **Fact 2.2.** $I(X,Y|Z) = H(X|Z) - H(X|Y,Z)$.

▶ **Fact 2.3** (Chain rule for conditional mutual information)**.** $I(X_1,X_2;Y|Z) = I(X_1;Y|Z) + I(X_2;Y|X_1,Z)$.

▶ **Fact 2.4** (Data processing inequality)**.** *If $I(X;Z|Y,W) = 0$, then $I(X;Y|W) \geq I(X;Z|W)$.*

▶ **Fact 2.5.** *If $I(X;W|Y,Z) = 0$, then $I(X;Y|Z) \geq I(X;Y|Z,W)$.*

▶ **Definition 2.6** (Kullback–Leibler divergence)**.** The *Kullback–Leibler (KL) divergence* between two distributions μ, ν is $D_{\mathrm{KL}}(\mu \parallel \nu) = \sum_x \mu(x) \log \frac{\mu(x)}{\nu(x)}$.

▶ **Fact 2.7** (Gibbs' inequality)**.** *For every distributions μ and ν, $D_{\mathrm{KL}}(\mu \parallel \nu) \geq 0$.*

▶ **Fact 2.8.** *For any distribution μ on $\mathcal{X} \times \mathcal{Y}$ with marginals μ_X and μ_Y, the mutual information of the random variables $(A,B) \sim \mu$ satisfies $I(A;B) = D(\mu \parallel \mu_X \mu_Y)$.*

▶ **Fact 2.9** (Log-sum inequality)**.** *Let $n \in \mathbb{N}$ and $a_1,\ldots,a_n,b_1,\ldots,b_n$ be non-negative real numbers. Define $A := \sum_{i=1}^n a_i$ and $B := \sum_{i=1}^n b_i$. Then, $\sum_{i=1}^n a_i \log(a_i/b_i) \geq A \log(A/B)$.*

▶ **Definition 2.10** (Information cost)**.** Let μ be a distribution with support $\{0,1\}^n \times \{0,1\}^n$ and let $(X,Y) \sim \mu$ where X is Alice's input and Y is Bob's input. The *information cost* of a protocol Π with respect to μ is defined by $\mathrm{IC}_\mu(\Pi) := I_\mu(\Pi(X,Y);X|Y) + I_\mu(\Pi(X,Y);Y|X)$.

▶ **Definition 2.11** (Prior-free information complexity)**.** Let $f : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a function and let $\epsilon > 0$. The *prior-free information complexity* of f with error rate ϵ is defined by $\mathrm{IC}(f,\epsilon) := \min_\Pi \max_\mu \mathrm{IC}_\mu(\Pi)$ where Π ranges over all protocols computing f with error probability at most ϵ on each input pair in $\{0,1\}^n \times \{0,1\}^n$ and μ ranges over all distributions with support $\{0,1\}^n \times \{0,1\}^n$.

▶ Remark. Braverman [5] distinguished between internal information measures that quantify the amount of information that Alice and Bob reveal to each other and external information measures that quantify the amount of information that Alice and Bob reveal to an external observer. Definitions 2.10 and 2.11 refer to the *internal* information cost and *internal* prior-free information complexity respectively.

3 Lower Bound for the Small-error Regime

In this section, we complete the proof of Theorem 1.2, giving an unconditional lower bound on the information complexity of $\textsc{Ham}_{n,d}$. In fact, we do more: we show that the same information complexity lower bound holds even for protocols that receive the additional promise that every block of n/d coordinates in $[n]$ contains exactly 1 or 3 coordinates on which x and y differ. Furthermore, we show that our information complexity lower bound holds under the distribution where we choose the inputs x and y uniformly at random from all such pairs of inputs that have Hamming distance exactly 1 on each block.

The proof has two main components. The first is our lower bound on the information complexity of the $\textsc{Ham}_{n,1\text{vs}.3}$ function, which is the more technically challenging component of the proof and which we defer to the next subsection. The second is a direct sum theorem for information complexity. In order to state this theorem, we must first introduce a bit more notation. We use $[n]$ to denote the set $\{1,\dots,n\}$. For $X = X_1X_2\cdots X_n \in \mathcal{X}^n$ and $i < k < n$, let $X_{[k]}$ and $X_{[i:k]}$ denote the strings $X_1\cdots X_k$ and $X_i\cdots X_k$ respectively. For $i \in [n]$, we use e_i to denote the n-bit string $z \in \{0,1\}^n$ with $z_i = 1$ and all other bits $z_j = 0$.

▶ **Definition 3.1** (Composed function)**.** The *composition* of the functions $f : \{0,1\}^k \to \{0,1\}$ and $g : \mathcal{X}\times\mathcal{Y} \to \{0,1\}$ is the function $f\circ g : \mathcal{X}^k\times\mathcal{Y}^k \to \{0,1\}$ defined by $(f\circ g)(x,y) = f\big(g(x_1,y_1),\dots,g(x_k,y_k)\big)$.

▶ **Definition 3.2.** For a vector $x \in \mathcal{X}^k$, an index $j \in [k]$, and an element $u \in \mathcal{X}$, define $x_{j\leftarrow u}$ to be the vector in $\mathcal{X}^k$ obtained by replacing the jth coordinate of x with u.

▶ **Definition 3.3** (Collapsing distributions)**.** A distribution μ over $\mathcal{X}\times\mathcal{Y}$ is a *collapsing distribution* for the composed function $f\circ g : \mathcal{X}^k\times\mathcal{Y}^k \to \{0,1\}$ if every point (x,y) in the support of μ, every $j \in [k]$, and every $(u,v) \in \mathcal{X}\times\mathcal{Y}$ satisfy $f\circ g(x_{j\leftarrow u}, y_{j\leftarrow v}) = g(u,v)$.

We use the following direct-sum theorem, which is essentially due to Bar-Yossef et al. [1] and to Braverman and Rao [6]. We include the proof for the convenience of the reader.

▶ **Theorem 3.4** (Direct-sum theorem)**.** *Let μ^k be a collapsing distribution for the composed function $f\circ g : \mathcal{X}^k\times\mathcal{Y}^k \to \{0,1\}$. For every $\epsilon \geq 0$, $\mathrm{IC}_{\mu^k}(f\circ g,\epsilon) \geq k\,\mathrm{IC}_\mu(g,\epsilon)$.*

Proof. Consider an ϵ-error protocol P for $f\circ g$ with optimal information cost over μ^k. Let $\Pi(x,y)$ be a random variable (over the private randomness of the protocol) denoting the transcript of the protocol on inputs $x,y \in \mathcal{X}^k\times\mathcal{Y}^k$. By the optimality of P and two applications of the chain rule for mutual information in opposite directions,

$$\begin{aligned}\mathrm{IC}_{\mu^k}(f\circ g,\epsilon) &= I(X;\Pi(X,Y)\mid Y) + I(Y;\Pi(X,Y)\mid X)\\ &= \sum_{i=1}^{k} I(X_i;\Pi(X,Y)\mid Y, X_{[i-1]}) + I(Y_i;\Pi(X,Y)\mid X, Y_{[i+1,k]}).\end{aligned}$$

Since $I(X_i;Y_{[i-1]}\mid X_{[i-1]},Y_{[i,k]}) = 0$, we have $I(X_i;\Pi(X,Y)\mid Y,X_{[i-1]}) \geq I(X_i;\Pi(X,Y)\mid X_{[i-1]},Y_{[i,k]})$. Similarly, $I(Y_i;\Pi(X,Y)\mid X,Y_{[i+1,k]}) \geq I(Y_i;\Pi(X,Y)\mid X_{[i]},Y_{[i+1,k]})$. So

$$\mathrm{IC}_{\mu^k}(f\circ g,\epsilon) \geq \sum_{i=1}^{k} I(X_i;\Pi(X,Y)\mid X_{[i-1]}Y_{[i,k]}) + I(Y_i;\Pi(X,Y)\mid X_{[i]}Y_{[i+1,k]}).$$

To complete the proof, we want to show that each summand is the information cost of an ϵ-error protocol for g over μ. Fix an index $i \in [k]$. Let P_i^* be a protocol that uses the

public randomness to draw $X'_1, \ldots, X'_{i-1}$ from the marginal of μ on $\mathcal{X}$ and $Y'_{i+1}, \ldots, Y'_k$ from the marginal of μ on $\mathcal{Y}$. Alice draws $X'_{i+1}, \ldots, X'_k$ using her private randomness so that $(X'_{i+1}, Y'_{i+1}), \ldots, (X'_k, Y'_k) \sim \mu$. Similarly, Bob uses his private randomness to draw $Y'_1, \ldots, Y'_{i-1}$ such that $(X'_1, Y'_1), \ldots, (X'_{i-1}, Y'_{i-1}) \sim \mu$. They then set $X'_i \leftarrow X_i$ and $Y'_i \leftarrow Y_i$. The protocol P_i^* then simulates P on (X', Y') and returns the value of $f \circ g(X', Y')$. Since μ^k is a collapsing distribution, $g(X_i, Y_i) = f \circ g(X', Y')$ and P_i^* is a valid ϵ-error protocol for g. In turn, this implies that

$$\begin{aligned}\mathrm{IC}_{\mu^k}(f \circ g, \epsilon) &\geq \sum_{i=1}^{k} I(X_i; \Pi(X,Y) \mid X_{[i-1]} Y_{[i,k]}) + I(Y_i; \Pi(X,Y) \mid X_{[i]} Y_{[i+1,k]}) \\ &\geq \sum_{i=1}^{k} \mathrm{IC}_{\mu}(g, \epsilon) = k \, \mathrm{IC}_{\mu}(g, \epsilon).\end{aligned}$$

◀

Let μ be the uniform distribution on pairs $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ at Hamming distance one from each other. In the following subsection, we show that every protocol for $\textsc{Ham}_{n,1\text{vs}.3}$ must have information complexity $\Omega(\min\{\log n, \log \frac{1}{\epsilon}\})$ under this distribution. We can then apply the direct sum theorem to complete the proof of Theorem 1.2.

Proof of Theorem 1.2. Any protocol for $\textsc{Ham}_{n,d}$ also is a valid protocol for the composed function $\mathrm{AND}_d \circ \textsc{Ham}_{n/d,1\text{vs}.3}$. So for every $\epsilon \geq 0$,

$$\mathrm{IC}(\textsc{Ham}_{n,d}, \epsilon) \geq \mathrm{IC}(\mathrm{AND}_d \circ \textsc{Ham}_{n/d,1\text{vs}.3}, \epsilon).$$

Let μ be the uniform distribution on pairs $(x, y) \in \{0,1\}^{n/d} \times \{0,1\}^{n/d}$ with Hamming distance 1. By definition, $\mathrm{IC}(\mathrm{AND}_d \circ \textsc{Ham}_{n/d,1\text{vs}.3}, \epsilon) \geq \mathrm{IC}_{\mu^d}(\mathrm{AND}_d \circ \textsc{Ham}_{n/d,1\text{vs}.3}, \epsilon)$. Moreover, since the support of μ is on pairs x, y at Hamming distance 1 from each other, μ^d is a collapsing distribution for $AND_d \circ \textsc{Ham}_{n/d,1\text{vs}.3}$. So by Theorem 3.4,

$$\mathrm{IC}_{\mu^d}(\mathrm{AND}_d \circ \textsc{Ham}_{n/d,1\text{vs}.3}, \epsilon) \geq d \, \mathrm{IC}_{\mu}(\textsc{Ham}_{n/d,1\text{vs}.3}, \epsilon)$$

and the theorem follows from Lemma 1.3. ◀

3.1 Proof of Lemma 1.3

In this section, we give a lower bound on the information complexity of protocols for $\textsc{Ham}_{n,1\text{vs}.3}$ under the distribution μ that is uniform over the pairs of vectors $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ at Hamming distance 1 from each other.

▶ **Fact 3.5** (Rectangle bound [1]). *For any protocol whose transcript on inputs x, y (resp., x', y') is the random variable $\Pi(x, y)$ (resp., $\Pi(x', y')$) and for any possible transcript t,*

$$\Pr[\Pi(x,y) = t] \Pr[\Pi(x',y') = t] = \Pr[\Pi(x,y') = t] \Pr[\Pi(x',y) = t].$$

▶ **Fact 3.6** (Extension of Gibbs' inequality). *For every distributions μ and ν on $\mathcal{X}$, and every subset $S \subseteq \mathcal{X}$, $\sum_{x \in S} \mu(x) \log \frac{\mu(x)}{\nu(x)} \geq \ln 2 \, (\mu(S) - \nu(S))$.*

Proof. Using the inequality $\log x \leq \ln 2 (x - 1)$, we obtain

$$\sum_{x \in S} \mu(x) \log \frac{\mu(x)}{\nu(x)} = -\sum_{x \in S} \mu(x) \log \frac{\nu(x)}{\mu(x)} \geq \sum_{x \in S} \mu(x) \ln 2 \, (1 - \frac{\nu(x)}{\mu(x)}) \geq \ln 2 \, (\mu(S) - \nu(S)).$$ ◀

▶ **Lemma 3.7.** *Let Π be a randomized protocol and let T be the set of all possible transcripts of Π. Let μ be the uniform distribution on pairs $(x,y) \in \{0,1\}^n \times \{0,1\}^n$ at Hamming distance 1 from each other. Then*

$$IC_\mu(\Pi(X,Y)) = \mathop{\mathbb{E}}_{z\in\{0,1\}^n, i\in[n]} \sum_{t\in T} \Pr[\Pi(z\oplus e_i, z) = t] \log \frac{\Pr[\Pi(z\oplus e_i, z) = t]}{\mathbb{E}_{j,\ell\in[n]} \Pr[\Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell) = t]}.$$

Proof. The mutual information of X and $\Pi(X,Y)$ given Y satisfies

$$\begin{aligned}
&I(X;\Pi(X,Y) \mid Y)\\
&\quad= \mathop{\mathbb{E}}_{y}[I(X;\Pi(X,y) \mid Y = y)]\\
&\quad= \mathop{\mathbb{E}}_{y}[D_{\mathrm{KL}}(X,\Pi(X,y) \parallel X,\Pi(X',y))]\\
&\quad= \mathop{\mathbb{E}}_{y}\left[\sum_{x\in\{0,1\}^n}\sum_{t\in T} \Pr[X=x]\Pr[\Pi(x,y)=t]\log\frac{\Pr[X=x]\Pr[\Pi(x,y)=t]}{\Pr[X=x]\Pr[\Pi[X',y]=t]}\right]\\
&\quad= \mathop{\mathbb{E}}_{z,i}\left[\sum_{t\in T}\Pr[\Pi(z\oplus e_i,z)=t]\log\frac{\Pr[\Pi(z\oplus e_i,z)=t]}{\mathbb{E}_{\ell\in[n]}\Pr[\Pi(z\oplus e_\ell,z)=t]}\right].
\end{aligned}$$

Similarly,

$$I(Y;\Pi(X,Y) \mid X) = \mathop{\mathbb{E}}_{z,i}\left[\sum_{t\in T}\Pr[\Pi(z\oplus e_i,z)=t]\log\frac{\Pr[\Pi(z\oplus e_i,z)=t]}{\mathbb{E}_{\ell\in[n]}\Pr[\Pi(z\oplus e_i,z\oplus e_i\oplus e_j)=t]}\right].$$

Summing those two expressions, we obtain

$$IC_\mu(\Pi(X,Y)) = \mathop{\mathbb{E}}_{z,i}\left[\sum_{t\in T}\Pr[\Pi(z\oplus e_i,z)=t] \log\frac{\Pr[\Pi(z\oplus e_i,z)=t]^2}{\mathbb{E}_{j,\ell\in[n]}\Pr[\Pi(z\oplus e_\ell,z)=t]\Pr[\Pi(z\oplus e_i,z\oplus e_i\oplus e_j)=t]}\right].$$

By the rectangle bound (Fact 3.5),

$$\Pr[\Pi(z\oplus e_\ell,z)=t]\Pr[\Pi(z\oplus e_i,z\oplus e_i\oplus e_j)=t] = \Pr[\Pi(z\oplus e_i,z)=t]\Pr[\Pi(z\oplus e_\ell,z\oplus e_i\oplus e_j)=t]$$

and the lemma follows. ◀

Proof of Lemma 1.3. Fix any ϵ-error protocol for $\textsc{Ham}_{n,1\text{vs}.3}$. Let $\Pi(x,y)$ denote (a random variable representing) its transcript on inputs x,y. Let T^1 denote the set of transcripts for which the protocol outputs 1. By Lemma 3.7 and the extended Gibbs' inequality (Fact 3.6),

$$\mathrm{IC}_\mu(\Pi(X,Y)) \geq \mathop{\mathbb{E}}_{z\in\{0,1\}^n, i\in[n]} \sum_{t\in T^1} \Pr[\Pi(z\oplus e_i,z)=t] \log\frac{\Pr[\Pi(z\oplus e_i,z)=t]}{\mathbb{E}_{j,\ell\in[n]}\Pr[\Pi(z\oplus e_i\oplus e_j,z\oplus e_\ell)=t]} - \ln 2\,.$$

The correctness of the protocol guarantees that when i,j,ℓ are all disjoint, then $\sum_{t\in T^1}\Pr[\Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell)=t] \leq \epsilon$. For any $z\in\{0,1\}^n$ and $i\in[n]$, the probability that i,j,ℓ are all disjoint is $(n-1)(n-2)/n^2 > 1-3/n$. Therefore,

$$\sum_{t\in T^1}\mathop{\mathbb{E}}_{j,\ell\in[n]}\Pr[\Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell)=t] \leq 3/n+\epsilon$$

and by the log-sum inequality and the fact that $x \log_2(x) \geq -0.6$ for all $x \in [0,1]$,

$$\begin{aligned}\mathrm{IC}_\mu(\Pi(X,Y)) &\geq \Pr[\Pi(z \oplus e_i, z) \in T^1] \log \frac{\Pr[\Pi(z \oplus e_i, z) \in T^1]}{\mathbb{E}_{j,\ell} \Pr[\Pi(z \oplus e_i \oplus e_j, z \oplus e_\ell) \in T^1]} \\ &\geq (1-\epsilon) \log \frac{1-\epsilon}{3/n+\epsilon} - \ln 2 \geq (1-\epsilon) \log \frac{1}{3/n+\epsilon} - O(1)\,. \end{aligned}$$

◀

4 Conditional Lower Bound

In this section, we prove Theorem 1.6. We will need the following notion of information complexity.

▶ **Definition 4.1** (Information complexity with average-case abortion and error)**.** Let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$. Then, $\mathrm{IC}_{\mu,\delta,\epsilon}(f|\nu)$ is the minimum conditional information cost of a randomized protocol that computes f with abortion probability at most δ and error probability at most ϵ, where the probabilities are taken over both the internal (public and private) randomness of the protocol Π and over the randomness of the distribution μ.

We now give the slight generalization of the MWY theorem that we will use to prove Theorem 1.6.

▶ **Theorem 4.2** (Slight generalization of the direct-sum theorem of [16])**.** *Let $X \in \mathcal{X}$, $Y \in \mathcal{Y}$ and λ be a distribution on (X,Y,D) with marginals μ over (X,Y) and ν over D such that for every value d of D, X and Y are conditionally independent given $D = d$. For any $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, $k \in \mathbb{N}$ and $\epsilon \leq 1/3$, $\mathrm{IC}_{\mu^k,\epsilon}(f^k|\nu^k) = k \cdot \Omega(\mathrm{IC}_{\mu,O(\epsilon),O(\epsilon/k)}(f|\nu))$.*

Proof. See appendix A for the proof and the comparison to the direct-sum theorem of [16]. ◀

We will lower bound the information revealed by any protocol computing $\mathrm{HAM}_{n,1}$ with small error and abortion with respect to some hard input distribution. Here, the error and abortion probabilities are over both the hard input distribution and the public and private randomness of the protocol. We handle abortion probabilities and allow such average-case guarantees in order to be able to apply Theorem 4.2. We first define our hard input probability distribution. We define the distribution λ over tuples (B,D,Z,I,J,L,X,Y) as follows: To sample $(B,D,Z,I,J,L,X,Y) \sim \lambda$, we sample $B, D \in_R \{0,1\}$, $Z \in_R \{0,1\}^n$, $I,J,L \in_R [n]$ and:

- If $B = 0$,
  - If $D = 0$, set $(X,Y) = (Z, Z \oplus e_I)$.
  - If $D = 1$, set $(X,Y) = (Z \oplus e_I, Z)$.
- If $B = 1$,
  - If $D = 0$, set $(X,Y) = (Z \oplus e_I \oplus e_J, Z \oplus e_L)$.
  - If $D = 1$, set $(X,Y) = (Z \oplus e_L, Z \oplus e_I \oplus e_J)$.

We let μ be the marginal of λ over (X,Y) (and ν be the marginal of λ over (B,D,Z)). Note that conditioned on B, D and Z taking any particular values, X and Y are independent. That is, we have a mixture of product distributions. We will prove the following lemma (which is a stronger version of Lemma 1.7).

▶ **Lemma 4.3.** *Let Π be a randomized protocol that computes $\mathrm{HAM}_{n,1}$ with abortion probability at most δ and error probability at most ϵ, where the probabilities are taken over both the internal (public and private) randomness of the protocol Π and over the randomness of our*

marginal distribution μ. Let q and w be such that $4/q + 4(\delta+\epsilon)/w \leq 1$ and $w \leq 1$. Then, we have that

$$I((X,Y);\Pi(X,Y)|Z,D,B=0) \geq (1-\frac{4}{q}-\frac{4(\delta+\epsilon)}{w})\frac{(1-w)}{2}\log_2(\frac{1}{3/n+q\epsilon}) - O(1)\,. \quad (2)$$

For $\delta \leq 1/32$ and $\epsilon \leq 1/32$, setting $w = 16(\delta+\epsilon)$ and $q = 16$ in inequality (2) yields

$$\begin{aligned} I((X,Y);\Pi(X,Y)|Z,D,B) &= \Omega(I((X,Y);\Pi(X,Y)|Z,D,B=0)) \\ &= \Omega(\min(\log n, \log(1/\epsilon))) - O(1)\,. \end{aligned}$$

Given Lemma 4.3, we can now complete the proof of Theorem 1.6.

Proof of Theorem 1.6. Since $\textsc{Ham}_{n,d} = \textsc{Ham}_{n,n-d}$, it suffices to prove the bound for $d \leq n/2$. Applying Theorem 4.2 with $f = \textsc{Ham}_{n/d,1}$, $k = d$ and the distributions μ and ν given above, we get that

$$\mathrm{IC}_{\mu^d,\epsilon}((\textsc{Ham}_{n/d,1})^d|\nu^d) = d \cdot \Omega(\mathrm{IC}_{\mu,O(\epsilon),O(\epsilon/d)}(\textsc{Ham}_{n/d,1}|\nu)).$$

By Lemma 4.3, we also have that

$$\mathrm{IC}_{\mu,O(\epsilon),O(\epsilon/d)}(\textsc{Ham}_{n/d,1}|\nu) = \Omega(\min(\log(n/d), \log(d/\epsilon))) - O(1).$$

Hence,

$$\mathrm{IC}_{\mu^d,\epsilon}((\textsc{Ham}_{n/d,1})^d|\nu^d) = d \cdot \Omega(\min(\log(n/d), \log(d/\epsilon))) - O(d).$$

Using the assumption that $\textsc{Ham}_{n/d,1}$ is majority-hard, Theorem 1.6 now follows. ◀

Given Lemma 4.3, we can also complete the proof of Lemma 1.7.

Proof of Lemma 1.7. Let Π be a randomized protocol that computes $\textsc{Ham}_{n,1}$ with abortion probability at most δ and error probability at most ϵ, where the probabilities are taken over both the internal (public and private) randomness of the protocol Π and over the randomness of our marginal distribution μ. We have that

$$\begin{aligned} \mathrm{IC}_\mu(\Pi) &= I_\mu(\Pi(X,Y);X|Y) + I_\mu(\Pi(X,Y);Y|X) \\ &\overset{(a)}{\geq} I_\lambda(\Pi(X,Y);X|Y,D,B) + I_\lambda(\Pi(X,Y);Y|X,D,B) \\ &\geq \frac{1}{4}(I_\lambda(\Pi(X,Y);X|Y,D=1,B=0) + I_\lambda(\Pi(X,Y);Y|X,D=0,B=0)) \\ &= \frac{1}{4}(I_\lambda(\Pi(X,Y);X|Z,D=1,B=0) + I_\lambda(\Pi(X,Y);Y|Z,D=0,B=0)) \\ &= \frac{1}{2}I_\lambda(\Pi(X,Y);X|Z,D,B=0) \\ &\overset{(b)}{=} \Omega(\min(\log n, \log(1/\epsilon))) - O(1). \end{aligned}$$

where (a) follows from Fact 2.5 and the fact that $I(\Pi(X,Y);(D,B)|X,Y) = 0$ and (b) follows from Lemma 4.3. ◀

4.1 Proof of Lemma 4.3

We start by sketching the idea of the proof of Lemma 4.3 before giving the full proof. We first note that the conditional information cost that we want to lower bound can be expressed as an average, over a part of the input distribution, of a quantity that still carries the randomness of the protocol. We show that most distance-1 input pairs are computed correctly and have an expected error probability over their distance-3 "cousin pairs"[3] of at most $O(\epsilon)$. We can thus average over only such distance-1 input pairs at the cost of a multiplicative constant-factor decrease in the lower bound. At this point, the remaining randomness is due solely to the protocol. It turns out that we can deal with the corresponding quantity in a similar way to how we dealt with the randomness in the proof of Lemma 1.3, i.e., using the extended Gibbs' inequality and the log-sum inequality. We now give the full proof.

Proof of Lemma 4.3. Let T be the set of all possible transcripts of Π. By Lemma 3.7, we have that[4]

$$\begin{aligned} &I((X,Y);\Pi|Z,D,B=0) \\ &\quad= \frac{1}{2}\mathop{\mathbb{E}}_{z\in\{0,1\}^n,i\in[n]}\sum_{t\in T}\Pr[\Pi(z\oplus e_i,z)=t]\log\frac{\Pr[\Pi(z\oplus e_i,z)=t]}{\mathbb{E}_{j,\ell\in[n]}\Pr[\Pi(z\oplus e_i\oplus e_j,z\oplus e_\ell)=t]} \\ &\quad= \frac{1}{2}\mathop{\mathbb{E}}_{z\in\{0,1\}^n,i\in[n]}\kappa_{z,i} \end{aligned}$$

with

$$\kappa_{z,i} := \sum_{t\in T}\Pr[\Pi(z\oplus e_i,z)=t]\log\frac{\Pr[\Pi(z\oplus e_i,z)=t]}{\mathbb{E}_{j,\ell\in[n]}\Pr[\Pi(z\oplus e_i\oplus e_j,z\oplus e_\ell)=t]}.$$

By the log-sum inequality, we have:

▶ **Fact 4.4.** *For every* $(z,i)\in\{0,1\}^n\times[n]$, $\kappa_{z,i}\geq 0$.

Let q and w be such that $4/q+4(\delta+\epsilon)/w\leq 1$ and $w\leq 1$.

▶ **Definition 4.5** (Nice (z,i)-pairs)**.** A pair $(z,i)\in\{0,1\}^n\times[n]$ is said to be *nice* if it satisfies the following two conditions:

1. $\Pr_{\Pi,j,l\in[n]}[\Pi(z\oplus e_i\oplus e_j,z\oplus e_\ell)\neq \textsc{Ham}_{n,1}(z\oplus e_i\oplus e_j,z\oplus e_\ell)$ and $\Pi(z\oplus e_i\oplus e_j,z\oplus e_\ell)$ does not abort] is at most $q\epsilon$.
2. $\Pr_\Pi[\Pi(z\oplus e_i,z)]\neq \textsc{Ham}_{n,1}(z\oplus e_i,z)]\leq w$

The following lemma shows that most (z,i)-pairs are nice:

▶ **Lemma 4.6.** *The fraction of pairs* $(z,i)\in\{0,1\}^n\times[n]$ *that are nice is at least* $1-4/q-4(\delta+\epsilon)/w$.

[3] For a distance-1 input pair $(z\oplus e_i,z)$, its distance-3 "cousin pairs" are those of the form $(z\oplus e_i\oplus e_j,z\oplus e_\ell)$ for $j,\ell\in[n]$. Note that this step uses the two-sided nature of our new distribution.

[4] Note that given $B=0$, (X,Y) is a uniformly-random distance-1 pair. Thus, $I((X,Y);\Pi(X,Y)|Z,D,B=0)$ is equal to the internal information complexity $\mathrm{IC}_\mu(\Pi(X,Y))$ in Lemma 3.7 up to a multiplicative factor of 2.

Proof of Lemma 4.6. We have that

$$\begin{aligned}
&\mathop{\mathbb{E}}_{z,i}[\Pr_{\Pi,j,l}[\Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell)\neq \textsc{Ham}_{n,1}(z\oplus e_i\oplus e_j, z\oplus e_\ell)\\
&\qquad\qquad\qquad\qquad\qquad\qquad \text{and } \Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell) \text{ does not abort}]]\\
=\ &\Pr_{z,i,\Pi,j,l}[\Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell)\neq \textsc{Ham}_{n,1}(z\oplus e_i\oplus e_j, z\oplus e_\ell)\\
&qquad\qquad\qquad\qquad\qquad\qquad \text{and } \Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell) \text{ does not abort}]\\
\leq\ &4\Pr_{\Pi,(x,y)\sim\mu}[\Pi(x,y)\neq \textsc{Ham}_{n,1}(x,y) \text{ and } \Pi(x,y) \text{ does not abort}]\\
\leq\ &4\epsilon.
\end{aligned}$$

Thus, by Markov's inequality, the fraction of (z,i)-pairs for which

$$\begin{aligned}
\Pr_{\Pi,j,l}[&\Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell)\neq \textsc{Ham}_{n,1}(z\oplus e_i\oplus e_j, z\oplus e_\ell)\\
&\qquad\qquad \text{and } \Pi(z\oplus e_i\oplus e_j, z\oplus e_\ell) \text{ does not abort}] > q\epsilon
\end{aligned}$$

is at most $4/q$. Moreover, we have that

$$\begin{aligned}
\mathop{\mathbb{E}}_{z,i}[\Pr_{\Pi}[\Pi(z\oplus e_i, z)\neq \textsc{Ham}_{n,1}(z\oplus e_i, z)]] &= \Pr_{\Pi,z,i}[\Pi(z\oplus e_i, z)\neq \textsc{Ham}_{n,1}(z\oplus e_i, z)]\\
&\leq 4\Pr_{\Pi,(x,y)\sim\mu}[\Pi(x,y)\neq \textsc{Ham}_{n,1}(x,y)]\\
&\leq 4(\delta+\epsilon).
\end{aligned}$$

Applying Markov's inequality once again, we get that the fraction of (z,i)-pairs for which

$$\Pr_{\Pi}[\Pi(z\oplus e_i, z)\neq \textsc{Ham}_{n,1}(z\oplus e_i, z)]\geq w$$

is at most $4(\delta+\epsilon)/w$. By the union bound, we conclude that the fraction of (z,i)-pairs that are nice is at least $1-4/q-4(\delta+\epsilon)/w$. ◀

Let $N\subseteq\{0,1\}^n\times[n]$ be the set of all nice (z,i)-pairs. Using the fact that $\kappa_{z,i}\geq 0$ for all z and i (Fact 4.4), we get that:

$$I((X,Y);\Pi(X,Y)|Z,D,B=0)\geq\frac{1}{2}\frac{|N|}{n2^n}\mathop{\mathbb{E}}_{(z,i)\in N}\left[\kappa_{z,i}\right]. \tag{3}$$

We have the following lemma:

▶ **Lemma 4.7.** *For every $(z,i)\in N$, $\kappa_{z,i}\geq(1-w)\log_2(\frac{1}{3/n+q\epsilon})-O(1)$.*

Proof of Lemma 4.7. Fix $(z,i)\in N$. Let $T^{(=1)}\subseteq T$ be the set of all transcripts that declare the input pair to be at distance 1. Using the extended Gibbs' inequality (Fact 3.6),

$$\kappa_{z,i}=\sum_{t\in T^{(=1)}}\Pr[\Pi(z\oplus e_i,z)=t]\log\frac{\Pr[\Pi(z\oplus e_i,z)=t]}{\mathbb{E}_{j,\ell\in[n]}\Pr[\Pi(z\oplus e_i\oplus e_j,z\oplus e_\ell)=t]}-\ln 2.$$

Using the log-sum inequality, Definition 4.5 and the fact that $x\log_2(x)\geq -0.6$ for all $x\in[0,1]$, we have that

$$\kappa_{z,i}\geq(1-w)\log_2(\frac{1-w}{3/n+q\epsilon})-\ln 2=(1-w)\log_2(\frac{1}{3/n+q\epsilon})-O(1).$$ ◀

Using Lemma 4.7 and Equation (3), we get

$$I((X,Y);\Pi(X,Y)|Z,D,B=0) \geq \frac{|N|}{n2^n}\frac{(1-w)}{2}\log_2(\frac{1}{3/n+q\epsilon}) - O(1)$$
$$\geq (1-\frac{4}{q}-\frac{4(\delta+\epsilon)}{w})\frac{(1-w)}{2}\log_2(\frac{1}{3/n+q\epsilon}) - O(1),$$

where the last inequality follows from Lemma 4.6. The second part of Lemma 4.3 follows from that the fact that

$$I((X,Y);\Pi(X,Y)|Z,D,B) = \frac{1}{2}\Big(I((X,Y);\Pi(X,Y)|Z,D,B=0) + I((X,Y);\Pi(X,Y)|Z,D,B=1)\Big).$$ ◀

5 Upper Bounds on the Complexity of Hamming Distance

5.1 Information Complexity Upper Bound

Algorithm 1 Protocol for $\text{HAM}_{n,d}$

Input. Alice is given $x \in \{0,1\}^n$ and Bob is given $y \in \{0,1\}^n$.
Parameters. $\epsilon \geq 0$, shared random string r.
Output. $\text{HAM}_{n,d}(x,y)$.

1: Alice and Bob use r to define a random k-partition P of $[n]$.
2: Alice sets $a \leftarrow h_P(x)$.
3: Bob sets $b \leftarrow h_P(y)$.
4: Alice and Bob initialize $c = 0$.
5: **for** $i = 1, \ldots, n$ **do**
6: Alice and Bob exchange a_i and b_i.
7: If $a_i \neq b_i$, they both update $c \leftarrow c + 1$.
8: If $c > d$, **return** 0.
9: **end for**
10: **return** 1.

In this section, we describe and analyze the protocol that establishes the upper bound on the information complexity of $\text{HAM}_{n,d}$ stated in Proposition 1.1. The protocol is described in Protocol 1. The analysis of the protocol relies on some basic inequalities that follow from a simple balls-and-bins lemma.

▶ **Definition 5.1** (Dot product). The *dot product* between vectors in $\{0,1\}^n$ is defined by setting $x \cdot y = \sum_{i=1}^n x_i y_i \pmod 2$.

▶ **Definition 5.2** (Random partition). For any $k < n$, a *random k-partition* P of $[n]$ is obtained by defining k sets $S_1, \ldots, S_k$ and putting each element $i \in [n]$ in one of those sets independently and uniformly at random. For $k \geq n$, we simply define P to be the complete partition $\{1\}, \ldots, \{n\}$ of $[n]$. We associate the partition P with a family of k elements $\alpha_1, \ldots, \alpha_k$ in $\{0,1\}^n$ by setting the ith coordinate of α_j to 1 iff $i \in S_j$.

▶ **Definition 5.3** (Hashing operator). For any $k \leq n$, the *k-hashing operator* $h_P : \{0,1\}^n \to \{0,1\}^k$ corresponding to the partition $P = (\alpha_1, \ldots, \alpha_k)$ of $[n]$ is the map defined by $h_P : x \mapsto (x \cdot \alpha_1, \ldots, x \cdot \alpha_k)$.

▶ **Lemma 5.4.** *Fix $d \geq 1$. If we throw at least $d+1$ balls into $(d+2)^2/\delta$ buckets independently and uniformly at random, then the probability that at most d buckets contain an odd number of balls is bounded above by δ.*

Proof. Toss the balls one at a time until the number r of remaining balls and the number t of buckets that contain an odd number of balls satisfy $r + t \leq d + 2$. If we toss all the balls without this condition being satisfied, then in the end we have more than $d + 2 > d + 1$ buckets with an odd number of balls and the lemma holds. Otherwise, fix r, t be the values when the condition $r + t \leq d + 2$ is first satisfied. Since r decreases by 1 everytime we toss a ball and t can only go up or down by 1 for each ball tossed, and since originally $r \geq d + 1$, we have $d + 1 \leq r + t \leq d + 2$. This implies that $r \leq d + 2$, that $t \leq d + 2$ and that if each of the r remaining balls land in one of the $(d + 2)^2/\delta - t$ buckets that currently contain an even number of balls, the conclusion of the lemma hold. The probability that this event does not hold is at most

$$\begin{aligned} \frac{t}{(d+2)^2/\delta} + \frac{t+1}{(d+2)^2/\delta} + \cdots + \frac{t+r-1}{(d+2)^2/\delta} &\leq \frac{rt + r(r-1)/2}{(d+2)^2/\delta} \\ &\leq \delta \frac{(\frac{d+2}{2})^2 + (d+2)(d+1)/2}{(d+2)^2} \leq \delta\,. \end{aligned}$$ ◀

▶ **Corollary 5.5.** *For every $x, y \in \{0,1\}^n$, the hashes $a = h_P(x)$ and $b = h_P(y)$ corresponding to a random $((d+2)^2/\epsilon)$-partition P of $[n]$ satisfy $\textsc{Ham}_{n,d}(a,b) = \textsc{Ham}_{n,d}(x,y)$ with probability at least $1 - \epsilon$.*

Proof. Let $S \subseteq [n]$ denote the set of coordinates $i \in [n]$ on which $x_i \neq y_i$. The number of coordinates $j \in [(d+2)^2/\epsilon]$ on which $a_j \neq b_j$ corresponds to the number of parts of the random partition P that receive an odd number of coordinates from S. This number corresponds to the number of buckets that receive an odd number of balls when $|S|$ balls are thrown uniformly and independently at random. When $|S| \leq d$, at most d buckets can contain a ball (and thus an odd number of balls) and so the corollary always holds. When $|S| \geq d + 1$, then by Lemma 5.4, the number of parts with an odd number of is also at least $d + 1$ except with probability at most ϵ. ◀

We are now ready to complete the proof of Proposition 1.1.

Proof of Proposition 1.1. Let us first examine the correctness of the protocol. When $\epsilon < n/(d+2)^2$, the protocol never errs since the players output 1 only when they verify (deterministically) that their strings have Hamming distance at most d. When $\epsilon \geq n/(d+2)^2$, the protocol is always correct when $\textsc{Ham}_{(d+2)^2/\epsilon,d}(a,b) = \textsc{Ham}_{n,d}(x,y)$. This identity always holds when the Hamming distance of x and y is at most d. And when the Hamming distance of x and y is greater than d, the identity is satisfied with probability at least $1 - \epsilon$ by Corollary 5.4.

Let us now analyze the information cost of the protocol. Write $m = \min\{n, (d+2)^2/\epsilon\}$ to denote the length of the vectors a and b. Let $\Pi(x,y)$ denote the transcript of the protocol on inputs x, y. Let μ be any distribution on $\{0,1\}^n \times \{0,1\}^n$. Let (X,Y) be drawn from μ and define $A = h_P(X)$, $B = h_P(Y)$. By the data processing inequality, since $I(\Pi(X,Y); X \mid Y, A) = 0$, the mutual information of $\Pi(X,Y)$ and X given Y satisfies

$$I(\Pi(X,Y); X \mid Y) \leq I(\Pi(X,Y); A \mid Y) = I(\Pi(A,B); A \mid B).$$

Furthermore, with $d \log m$ bits we can identify the first d coordinates $i \in [m]$ for which $a_i \neq b_i$ and thereby completely determine $\Pi(A, B)$. So by Fact 2.1,

$$H(\Pi(X, Y) \mid Y) \leq d \log m.$$

The same argument also yields $I(\Pi(X, Y); Y \mid X) \leq d \log m$, showing that the information cost of the protocol is at most $2d \log m$. ◀

5.2 Communication Complexity

Huang et al. [13], building on previous results by Yao [26] and by Gavinsky et al. [11], showed that the randomized communication complexity of $\textsc{Ham}_{n,d}$ in the simultaneous message passing (SMP) model is bounded above by $R^{\|,\text{pub}}_{1/3}(\textsc{Ham}_{n,d}) = O(d \log d)$. We simplify their protocol and refine this analysis to give a general upper bound on the communication complexity for arbitrary values of ϵ.

▶ **Theorem 5.6.** *Fix $\epsilon > 0$. The randomized communication complexity of $\textsc{Ham}_{n,d}$ in the simultaneous message passing model is bounded above by*

$$R^{\|,\text{pub}}_{\epsilon}(\textsc{Ham}_{n,d}) = O(\min\{d \log n + \log 1/\epsilon, d \log d/\epsilon).$$

The proof of the theorem uses the following results.

▶ **Lemma 5.7.** $R^{\|,\text{pub}}_{\epsilon}(\textsc{Ham}_{n,d}) = O(d \log n + \log 1/\epsilon)$.

Proof. Alice and Bob can generate $q = \log \binom{n}{\leq d} + \log \frac{1}{\epsilon}$ random vectors $r_1, \ldots, r_q \in \{0,1\}^n$ and send the dot products $x \cdot r_1, \ldots, x \cdot r_q$ and $y \cdot r_1, \ldots, y \cdot r_q$ to the verifier, respectively. The verifier then returns 1 iff there is a vector $z \in \{0,1\}^n$ of Hamming weight at most d such that $x \cdot r_j = y \cdot r_j \oplus z \cdot r_j$ for every $j \in [q]$. When $\textsc{Ham}_(x, y) \leq d$, the verifier always returns 1 since in this case $x \cdot r_j = (y \oplus z) \cdot r_j = y \cdot r_j \oplus z \cdot r_j$ for some vector z of Hamming weight at most d. And for any $z \in \{0,1\}^n$, when $x \neq y \oplus z$, the probability that the identity $x \cdot r_j = y \cdot r_j \oplus z \cdot r_j$ holds for every $j \in [q]$ is 2^{-q}. So, by the union bound, the overall probability that the verifier erroneously outputs 1 is at most $\binom{n}{\leq d} 2^{-q} = \epsilon$. ◀

▶ **Lemma 5.8.** $R^{\|,\text{pub}}_{\epsilon}(\textsc{Ham}_{n,d}) \leq R^{\|,\text{pub}}_{\epsilon/2}(\textsc{Ham}_{(d+2)^2/\epsilon,d})$.

Proof. Consider the protocol where Alice and Bob use the shared random string to generate a $(d+2)^2/\epsilon$-hash of their inputs x, y and then apply the protocol for $\textsc{Ham}_{(d+2)^2/\epsilon,d}$ with error $\epsilon/2$. By Corollary 5.5, the probability that the hashed inputs a, b do not satisfy $\textsc{Ham}_{n,d}(a, b) = \textsc{Ham}_{n,d}(x, y)$ is at most $\frac{\epsilon}{2}$. The lemma follows from the union bound. ◀

We can now complete the proof of the theorem.

Proof of Theorem 5.6. When $\epsilon \leq d/n$, Alice and Bob simply run the protocol from the proof of Lemma 5.7. When $\epsilon > d/n$, Alice and Bob combine the protocol from the proof of Lemma 5.8 with the protocol from Lemma 5.7 (with the parameter n set to $(d+2)^2/\epsilon$). ◀

6 Applications and Extensions

6.1 Property Testing Lower Bounds

A Boolean property P is a subset of the set of functions mapping $\{0,1\}^n$ to $\{0,1\}$. A function f has property P if $f \in P$. Conversely, we say that the function f is ϵ-*far* from P if

$|\{x \in \{0,1\}^n : f(x) \neq g(x)\}| \geq \epsilon 2^n$ for every $g \in P$. A (q, ϵ, δ)-*tester* for P is a randomized algorithm A that, given oracle access to some function $f : \{0,1\}^n \to \{0,1\}$, queries the value of f on at most q elements from $\{0,1\}^n$ and satisfies two conditions:

1. When f has property P, A accepts f with probability at least $1 - \delta$.
2. When f is ϵ-far from P, A rejects f with probability at least $1 - \delta$.

The query complexity of the property P for given ϵ and δ parameters is the minimum value of q for which there is a (q, ϵ, δ)-tester for P. We denote this query complexity by $Q_{\epsilon,\delta}(P)$.

The two properties we consider in this section are k-linearity and k-juntas. The function f is *k-linear* iff it is of the form $f : x \mapsto \sum_{i \in S} x_i \pmod 2$ for some set $S \subseteq [n]$ of size $|S| = k$. (The k-linear functions are also known as *k-parity* functions.) The function f is a *k-junta* if there is a set $J = \{j_1, \ldots, j_k\} \subseteq [n]$ of coordinates such that the value of $f(x)$ is determined by the values of $x_{j_1}, \ldots, x_{j_k}$ for every $x \in \{0,1\}^n$.

The upper bound in Corollary 1.9 is from [3]. The proof is obtained via a simple reduction from the Hamming distance function, following the method introduced in [4].

▶ **Corollary 6.1** (Unconditional lower bound of Corollary 1.9). *Fix $0 < \delta < \frac{1}{3}$, $0 < \epsilon \leq \frac{1}{2}$, and $k \leq n/\log \frac{1}{\delta}$. Then $Q_{\epsilon,\delta}(k\text{-Linearity}) = \Omega(k \log \frac{1}{\delta})$ and $Q_{\epsilon,\delta}(k\text{-Juntas}) = \Omega(k \log \frac{1}{\delta})$.*

Proof. Consider the following protocol for the $\textsc{Ham}_{n,k}$ function. Alice takes her input $x \in \{0,1\}^n$ and builds the function $\chi_A : \{0,1\}^n \to \{0,1\}$ defined by $\chi_A : z \mapsto \sum_{i=1}^n x_i z_i \pmod 2$. Similarly, Bob builds the function χ_B from his input y by setting $\chi_B : z \mapsto \sum_{i=1}^n y_i z_i \pmod 2$. Notice that the bitwise XOR of the functions χ_A and χ_B satisfies

$$\chi_A \oplus \chi_B : z \mapsto \sum_{i=1}^{n} (x_i + y_i) z_i \pmod 2 = \sum_{i \in [n] : x_i \neq y_i} z_i \pmod 2.$$

The function $\psi := \chi_A \oplus \chi_B$ is ℓ-linear, where ℓ is the Hamming distance of x and y. When $\ell \leq k$, the function ψ is a k-junta; when $\ell > k$, then ψ is $\frac{1}{2}$-far from all k-juntas. Let Alice and Bob simulate a q-query tester for k-juntas on ψ by exchanging the values of $\chi_A(z)$ and $\chi_B(z)$ for every query z of the tester. If this tester succeeds with probability $1 - \delta$, the resulting protocol is a δ-error protocol for $\textsc{Ham}_{n,k}$ with communication cost at most $2q$. Therefore, by Theorem 1.2, $Q_{\epsilon,\delta}(k\text{-Juntas}) \geq R^{\text{pub}}_{\delta}(\textsc{Ham}_{n,k}) \geq \Omega(k \log \frac{1}{\delta})$.

The lower bound for $Q_{\epsilon,\delta}(k\text{-Linearity})$ is essentially the same except that we use the extra fact that the bound in Theorem 1.2 also holds even when we have the additional promise that the Hamming distance between x and y is either exactly d or greater than d. ◀

The proof of the conditional lower bounds of Corollary 1.9 is identical except that we appeal to the bound in Theorem 1.6 instead of the one in Theorem 1.2 in the conclusion of the proof.

6.2 Parity Decision Tree Complexity Lower Bounds

The proof of Corollary 1.10 is similar to the one in the last section. The details follow.

Proof of Corollary 1.10. Consider the following protocol for the $\textsc{Ham}_{n,d}$ function. Let $z = x \oplus y \in \{0,1\}^n$ denote the bitwise XOR of Alice's input x and Bob's input y. The Hamming weight of z is exactly the Hamming distance between x and y. Recall that a randomized parity decision tree of depth d is a distribution over deterministic parity decision trees that each have depth at most d. Alice and Bob can use their shared randomness to draw a tree T from this distribution. Since for every $S \subseteq [n]$, the parity of z on S, denoted z_S, satisfies $z_S = x_S \oplus y_S$, Alice and Bob can determine the path of z through T by exchanging

the parities x_S and y_S for each query of the parity of z on the subset $S \subseteq [n]$ of coordinates. So they can determine the value of $\textsc{Ham}_{n,d}$ with error at most ϵ using $2R_\epsilon^{\oplus}(\textsc{Weight}_{n,d})$ bits of communication. The bounds in Corollary 1.10 follow directly from Theorems 1.2 and 1.6. ◀

6.3 Symmetric XOR Functions

The key to the proof of Theorem 1.11 is the observation that the proof of Theorem 1.2 proves an even stronger statement: it shows that the same information complexity bound also holds for the $\textsc{Ham}_{n,d\text{vs.}d+2}$ promise version of the $\textsc{Ham}_{n,d}$ function.

▶ **Theorem 6.2** (Strengthening of Theorem 1.2). *For every $1 \leq d < n-1$ and every $0 \leq \epsilon < 1/2$,*

$$\mathrm{IC}(\textsc{Ham}_{n,d\text{vs.}d+2}, \epsilon) = \Omega(\min\{\log \tbinom{n}{d}, d \log(1/\epsilon)\}).$$

Proof. The proof is identical to that of Theorem 1.2. The only additional observation that we need to make is that in our argument, our choice of μ^k ensures that we only ever examine the behavior of the protocol on inputs of the $\mathrm{AND}_d \circ \textsc{Ham}_{n,1\text{vs.}3}$ function in which at most 1 of the d inputs to the $\textsc{Ham}_{n,1\text{vs.}3}$ function have Hamming weight 3. ◀

The proof of Theorem 1.11 follows immediately from Theorem 6.2.

Proof of Theorem 1.11. Consider any ϵ-error protocol P for the symmetric XOR function f. Let $d = \Gamma_{+2}(f)$. Then since $f(d) \neq f(d+2)$, P must distinguish between the cases where Alice and Bob's inputs have Hamming distance d from those where their inputs have Hamming distance $d+2$. Thus, the protocol P (or the protocol P' obtained by flipping the outputs of P) is an ϵ-error protocol for $\textsc{Ham}_{n,d\text{vs.}d+2}$ and so it must have information cost at least $\mathrm{IC}(\textsc{Ham}_{n,d\text{vs.}d+2}, \epsilon)$ and the bound follows from Theorem 6.2. ◀

7 Direct-sum Theorems for Hamming Distance

It was shown in [10] that, when the error rate is viewed as a parameter, the equality function violates the direct-sum theorem for randomized communication complexity in the following sense:

▶ **Definition 7.1.** We say that a function $f : \{0,1\}^m \times \{0,1\}^m \to \{0,1\}$ violates the direct-sum theorem for randomized communication complexity if

$$R_\epsilon^k(f^k) = o(kR_\epsilon(f))$$

where $R_\epsilon^k(f^k)$ denotes the randomized communication complexity of computing f such that on each tuple of k input pairs, the error probability on each input pair is at most ϵ.

Braverman [5] showed that his constant upper bound on the information complexity of EQ (which holds for any error rate $\epsilon \geq 0$) implies a different proof of the fact that EQ violates the direct-sum theorem for randomized communication complexity when $\epsilon = o(1)$ is viewed as a parameter. We next observe that our tight characterization of the information complexity of HD_1^m given in Proposition 1.1 and Theorem 1.2 implies that HD_1^m satisfies the direct-sum theorem for randomized communication complexity whenever $m = \Omega(\mathrm{poly}(1/\epsilon))$ and violates it otherwise (i.e., when $\log m = o(\log(1/\epsilon))$). This can be seen as a further indication of the qualitative difference between the information complexity of EQ and that of HD_1^m in the small error regime.

▶ **Proposition 7.2.** *HD_1^m satisfies the direct-sum theorem for randomized communication complexity whenever $m = \Omega(\text{poly}(1/\epsilon))$ and violates it otherwise (i.e., when $\log m = o(\log(1/\epsilon))$).*

Proof. We first recall the following theorem of Braverman [5]:

▶ **Theorem 7.3** ([5])**.** *For any function f and any error rate $\epsilon > 0$, $IC(f, \epsilon) = \lim_{k\to\infty} \frac{R_\epsilon^k(f^k)}{k}$.*

Applying Theorem 7.3 with $f = HD_1^m$, we get that that $R_\epsilon^k((HD_1^m)^k) = \Theta(kIC(HD_1^m, \epsilon))$. Proposition 1.1 and Theorem 1.2, we have that $IC(HD_1^m, \epsilon) = \Theta(\min(\log m, \log(1/\epsilon)))$. Hence, we get that

$$R_\epsilon^k((HD_1^m)^k) = \Theta(k \min(\log m, \log(1/\epsilon)))$$

On the other hand, we have that $R_\epsilon(HD_1^m) = \Omega(\log(1/\epsilon))$ [5]. So we conclude that

$$R_\epsilon^k((HD_1^m)^k) = \Theta(kR_\epsilon(HD_1^m))$$

whenever $m = \Omega(\text{poly}(1/\epsilon))$ and

$$R_\epsilon^k((HD_1^m)^k) = o(kR_\epsilon(HD_1^m))$$

whenever $\log m = o(\log(1/\epsilon))$. ◀

8 Low Information Protocols for $AND_k \circ \text{Ham}_{n/k,1}$ and $OR_k \circ \text{Ham}_{n/d,1}$

In this section, we give protocols for $AND_k \circ \text{HAM}_{n/k,1}$ and $OR_k \circ \text{HAM}_{n/k,1}$ with $O(k)$ information cost. For $AND_k \circ \text{HAM}_{n/k,1}$, the following theorem implies a protocol with $O(k)$ information cost for any constant error parameter $\epsilon > 0$.

▶ **Theorem 8.1.** *For any error parameter $\epsilon > 0$,*

$$\text{IC}(AND_k \circ \text{HAM}_{n/k,1}, \epsilon) = O(k \min(\log(n/k), \log(1/\epsilon))).$$

Proof. The description of the protocol is given below.

Algorithm 2 Protocol for $AND_k \circ \text{HAM}_{n/k,1}$

Input. Alice is given $x \in \{0,1\}^n$ and Bob is given $y \in \{0,1\}^n$

Output. $AND_k \circ \text{HAM}_{n/k,1}(x, y)$

1: Run in parallel k copies of Algorithm 1 for $\text{HAM}_{n/k,1}$ with error parameter ϵ on $(x^{(1)}, y^{(1)}), \ldots, (x^{(k)}, y^{(k)})$.
2: Declare $AND_k \circ \text{HAM}_{n/k,1}(x, y)$ to be 1 if and only if all the $(x^{(i)}, y^{(i)})$'s were declared to be at distance 1.

If $AND_k \circ \text{HAM}_{n/k,1}(x, y) = 1$, then all the $(x^{(i)}, y^{(i)})$'s are at distance 1. Since Algorithm 1 for $\text{HAM}_{n/k,1}$ always outputs the correct answer on distance-1 input pairs, each $(x^{(i)}, y^{(i)})$ will be declared to be at distance 1 and hence the above protocol will output the correct answer for $AND_k \circ \text{HAM}_{n/k,1}(x, y)$ (namely, 1) with probability 1. If $AND_k \circ \text{HAM}_{n/k,1}(x, y) = 0$, then there exists an $(x^{(i)}, y^{(i)})$ that is at distance 3. Then, the copy of Algorithm 1 for $\text{HAM}_{n/k,1}$

[5] This follows from the fact that $R_\epsilon(EQ) = \Omega(\log(1/\epsilon))$ and by padding.

running on $(x^{(i)}, y^{(i)})$ will declare this pair to be at distance 3 with probability at least $1-\epsilon$. Thus, the above protocol will output the correct answer for $AND_k \circ \text{HAM}_{n/k,1}(x,y)$ (namely, 0) with probability at least $1-\epsilon$. Fix a distribution μ on the input pair (X,Y) with support $\{0,1\}^{2n}$ and let $\mu^{(i)}$ denote the marginal of μ over $(X^{(i)}, Y^{(i)})$ for every $i \in [k]$. Denoting by Π the transcript of the above protocol, its information cost $\text{IC}_\mu(\Pi) := I_\mu(\Pi; X|Y) + I_\mu(\Pi; Y|X)$ is upper-bounded by the following lemma:

▶ **Lemma 8.2.** $\text{IC}_\mu(\Pi) = O(k \min(\log(n/k), \log(1/\epsilon)))$.

Proof. Denote by $\Pi^{(1)}, \ldots, \Pi^{(k)}$ the transcripts corresponding to the k parallel runs of Algorithm 1 for $\text{HAM}_{n/k,1}$ on the input pairs $(x^{(1)}, y^{(1)}), \ldots, (x^{(k)}, y^{(k)})$ respectively. Since $\Pi^{(1)}, \ldots, \Pi^{(k)}$ completely determine Π, we have that

$$\text{IC}_\mu(\Pi) = I_\mu(\Pi^{(1)}, \ldots, \Pi^{(k)}; X|Y) + I_\mu(\Pi^{(1)}, \ldots, \Pi^{(k)}; Y|X).$$

Since each of the protocols $\Pi^{(1)}, \ldots, \Pi^{(k)}$ - as well as Π - is completely symmetric with respect to Alice and Bob, it is enough to show that $I_\mu(\Pi^{(1)}, \ldots, \Pi^{(k)}; X|Y) = O(k \min(\log(n/k), \log(1/\epsilon)))$. By the chain rule for mutual information, we have that:

$$\begin{aligned}
I_\mu(\Pi^{(1)}, \ldots, \Pi^{(k)}; X|Y) &= \sum_{i=1}^{k} I_\mu(\Pi^{(i)}; X|Y, \Pi^{(<i)}) \\
&= \sum_{i=1}^{k} \sum_{j=1}^{k} I_\mu(\Pi^{(i)}; X^{(j)}|Y, \Pi^{(<i)}, X^{(<j)}) \\
&\overset{(a)}{=} \sum_{i=1}^{k} I_\mu(\Pi^{(i)}; X^{(i)}|Y, \Pi^{(<i)}, X^{(<i)}) \\
&\overset{(b)}{=} \sum_{i=1}^{k} I_{\mu^{(i)}}(\Pi^{(i)}; X^{(i)}|Y^{(i)}) \\
&\overset{(c)}{=} \sum_{i=1}^{k} O(\min(\log(n/k), \log(1/\epsilon))) \\
&= O(k \min(\log(n/k), \log(1/\epsilon)))
\end{aligned}$$

where (a) follows from $\Pi^{(i)}$ and $X^{(j)}$ being conditionally independent given $Y, \Pi^{(<i)}, X^{(<j)}$ for any $i \neq j \in [k]$, (b) follows from $(\Pi^{(i)}, X^{(i)})$ being conditionally independent of $Y^{(\neq i)}, \Pi^{(<i)}, X^{(<i)}$ given $Y^{(i)}$ and (c) follows from Proposition 1.1. ◀

The previous lemma implies that for constant ϵ, the information cost of protocol Π is $O(k)$. The following lemma notes that, in this case, even the communication complexity is $O(k)$:

▶ **Lemma 8.3.** *For constant ϵ, the communication complexity of Algorithm 2 is $O(k)$.*

Proof. Note that for constant ϵ, Theorem 5.6 implies that each run of Algorithm 1 has communication cost $O(1)$. Since Algorithm 2 performs k such calls to Algorithm 1, the communication cost of Algorithm 2 is hence $O(k)$. ◀

◀

▶ **Theorem 8.4.** *For every constant $\nu \in (0,1)$, $CC(OR_k \circ \text{HAM}_{n/k,1}, 1/k^\nu) = O(k)$.*

Algorithm 3 Algorithm for $OR_k \circ \text{HAM}_{n/k,1}$

Input. Alice is given $x \in \{0,1\}^n$ and Bob is given $y \in \{0,1\}^n$
Output. $OR_k \circ \text{HAM}_{n/k,1}(x,y)$

1: Let $c := \nu + 1$, $\eta := 1/4$, $t := c \log_2 k$, and $h := t/2$.
2: Mark all k input pairs $(x^{(1)}, y^{(1)}), \ldots, (x^{(k)}, y^{(k)})$ as distance-1 pairs.
3: Initialize the number u of inputs pairs that are marked to be at distance 1: $u = k$.
4: **for** $i = 1 : t$ **do**
5: Run in parallel u copies of Protocol 1 for $\text{HAM}_{n/k,1}$ with error parameter $\epsilon' = 1/2$ on each of the input pairs $(x^{(i)}, y^{(i)})$ that are still marked as distance-1 pairs.
6: If an input pair is declared to be at distance 3, mark it as a distance-3 pair.
7: If $i \leq h$ and the number u of input pairs that are still marked as distance-1 pairs is larger than $(1+\eta)k/2^i$, halt and declare $OR_k \circ \text{HAM}_{n/k,1}(x,y)$ to be 1.
8: **end for**
9: Declare $OR_k \circ \text{HAM}_{n/k,1}(x,y)$ to be 0 if and only if all the $(x^{(i)}, y^{(i)})$'s are marked as distance-3 pairs.

Proof. The description of the protocol is given in Algorithm 3.

If $OR_k \circ \text{HAM}_{n/k,1}(x,y) = 1$, then there is an input pair $(x^{(i)}, y^{(i)})$ (for some $i \in [k]$) that is at distance 1. Since Protocol 1 for $\text{HAM}_{n/k,1}$ always outputs the correct answer on distance-1 input pairs, $(x^{(i)}, y^{(i)})$ will be declared to be at distance 1 in each iteration and hence the above protocol will output the correct answer for $OR_k \circ \text{HAM}_{n/k,1}(x,y)$ (namely, 1) with probability 1. If $OR_k \circ \text{HAM}_{n/k,1}(x,y) = 0$, then the protocol outputs the correct answer with probablity at least $1 - \epsilon$ as shown by the following lemma:

▶ **Lemma 8.5.** *If $OR_k \circ \text{HAM}_{n/k,1}(x,y) = 0$, then the probability that the protocol outputs a wrong answer is at most $1/k^{c-1} + ke^{-\frac{\eta^2 k^{1-c/2}}{3}}$.*

Proof. If $OR_k \circ \text{HAM}_{n/k,1}(x,y) = 0$, all the $(x^{(i)}, y^{(i)})$'s are at distance 3. Conditioned on the fact that the protocol didn't halt and output 1 during the for loop, the probability that the protocol outputs an incorrect answer is, by the union bound, at most $k \times 1/2^t = 1/k^{c-1}$. To complete the proof, we now upper bound the probability that the protocol halts and outputs 1 during the for loop. Note that the expected number of input pairs that are marked as distance-1 pairs after the i-th iteration is $k/2^i$. By the Chernoff bound, the probability that after the i-th iteration, the number of distance-1 marked pairs is larger than $(1+\eta)k/2^i$ is at most

$$e^{-\eta^2 k/(3\times 2^i)} \leq e^{-\eta^2 k/(3\times 2^h)} = e^{-\frac{\eta^2 k^{1-c/2}}{3}}.$$

By the union bound, the probability that the algorithm halts and outputs 0 during the for loop is at most $ke^{-\frac{\eta^2 k^{1-c/2}}{3}}$. By another union bound, the probability that the protocol outputs an incorrect answer is at most $1/k^{c-1} + ke^{-\frac{\eta^2 k^{1-c/2}}{3}}$. ◀

▶ **Lemma 8.6.** *For any constant $c \in (1,2)$, the communication complexity of the above protocol is $O(1)$.*

Proof. Consider the execution of Protocol 3. For every $i \in [h]$, the number of calls to Protocol 1 is at most $k(1+\eta)/2^{i-1}$. For every $i \in \{h+1, \ldots, k\}$, the number of calls of

Protocol 1 is at most $k(1+\eta)/2^h$. Hence, the total number of calls to Protocol 1 is at most:

$$\sum_{i=1}^{h} \frac{k(1+\eta)}{2^{i-1}} + \frac{hk(1+\eta)}{2^h} \leq 2k(1+\eta) + \frac{ck(1+\eta)\log_2 k}{2^{\frac{ck\log_2 k}{2}+1}} = 2k(1+\eta) + \frac{c(1+\eta)}{2} k^{1-c/2}\log_2 k = \Theta(k)$$

where the last equality uses the fact that $c \in (1,2)$ is a constant. By Theorem 5.6, the communication cost of any run of Protocol 1 with noise rate $\epsilon' = 1/2$ is $O(1)$. Hence, the communication cost of Protocol 3 is $O(1)$. ◀

Using Lemma 8.5 (and the paragraph preceding it), Lemma 8.6 and the fact that $\nu = c-1$ is a constant in $(0,1)$, the statement of Theorem 8.4 now follows. ◀

Acknowledgments. The authors would like to thank Madhu Sudan for very helpful discussions. They also wish to thank the anonymous referees for much valuable feedback.
Eric Blais is supported by a Simons Postdoctoral Fellowship.

References

1 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. In *Proc. 43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 209–218, 2002.

2 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. In *STOC*, pages 67–76, 2010.

3 Eric Blais. Testing juntas nearly optimally. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 151–158. ACM, 2009.

4 Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012.

5 Mark Braverman. Interactive information complexity. In *Proc. 44th Annual ACM Symposium on the Theory of Computing*, 2012.

6 Mark Braverman and Anup Rao. Information equals amortized communication. In *FOCS*, pages 748–757, 2011.

7 Amit Chakrabarti and Oded Regev. An optimal lower bound on the communication complexity of Gap-Hamming-Distance. *SIAM Journal on Computing*, 41(5):1299–1317, 2012.

8 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.

9 Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

10 Tomas Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized communication complexity. *SIAM Journal on Computing*, 24(4):736–750, 1995.

11 Dmitry Gavinsky, Julia Kempe, and Ronald de Wolf. Quantum communication cannot simulate a public coin. *arXiv preprint quant-ph/0411051*, 2004.

12 Johan Håstad and Avi Wigderson. The randomized communication complexity of set disjointness. *Theory of Computing*, 3(1):211–219, 2007.

13 Wei Huang, Yaoyun Shi, Shengyu Zhang, and Yufan Zhu. The communication complexity of the hamming distance problem. *Inform. Process. Lett.*, 99:149–153, 2006.

14 Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Disc. Math.*, 5(4):547–557, 1992.

15 Iordanis Kerenidis, Sophie Laplante, Virginie Lerays, Jérémie Roland, and David Xiao. Lower bounds on information complexity via zero-communication protocols and applications. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 500–509. IEEE, 2012.

16 Marco Molinaro, David P Woodruff, and Grigory Yaroslavtsev. Beating the direct sum theorem in communication complexity with implications for sketching. In *SODA*, pages 1738–1756. SIAM, 2013.

17 Ryan O'Donnell. Hardness amplification within NP. *J. Comput. Syst. Sci.*, 69(1):68–94, 2004.

18 King F Pang and Abbas El Gamal. Communication complexity of computing the hamming distance. *SIAM Journal on Computing*, 15(4):932–947, 1986.

19 Ramamohan Paturi. On the degree of polynomials that approximate symmetric boolean functions (preliminary version). In *STOC*, pages 468–474, 1992.

20 Mert Sağlam and Gábor Tardos. On the communication complexity of sparse set disjointness and exists-equal problems. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 678–687. IEEE, 2013.

21 Alexander A Sherstov. The communication complexity of gap hamming distance. *Theory of Computing*, 8(1):197–208, 2012.

22 Thomas Vidick. A concentration inequality for the overlap of a vector on a large set, with application to the communication complexity of the gap-hamming-distance problem. *Chicago Journal of Theoretical Computer Science*, 1, 2012.

23 Emanuele Viola and Avi Wigderson. Norms, XOR lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(1):137–168, 2008.

24 David P Woodruff and Qin Zhang. Tight bounds for distributed functional monitoring. In *Proceedings of the 44th symposium on Theory of Computing*, pages 941–960. ACM, 2012.

25 Andrew C. Yao. Some complexity questions related to distributive computing. In *Proc. 11th Annual ACM Symposium on the Theory of Computing*, pages 209–213, 1979.

26 Andrew Chi-Chih Yao. On the power of quantum fingerprinting. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 77–81. ACM, 2003.

A Slight Generalization of the Direct-sum Theorem of [16]

We start by recalling the direct-sum theorem of Molinaro, Woodruff and Yaroslavtsev ([16]), which is stated in terms of the following notion of information complexity:

▶ **Definition 1.1** (MWY notion of information complexity with abortion)**.** Let $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ be a function. Then, $\mathrm{IC}_{\mu,\alpha,\delta,\epsilon}(f|\nu)$ is the minimum conditional information cost of a randomized protocol that with probability at least $1-\alpha$ gives a deterministic protocol that computes f with abortion probability at most δ with respect to μ and with conditional error probability given no abortion at most ϵ with respect to μ.

▶ **Theorem 1.2** ([16])**.** *Let $X \in \mathcal{X}$, $Y \in \mathcal{Y}$ and λ be a distribution on (X,Y,D) with marginals μ over (X,Y) and ν over D such that for every value d of D, X and Y are conditionally independent given $D = d$. For any $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, $k \in \mathbb{N}$ and $\delta \leq 1/3$,* $\mathrm{IC}_{\mu^k,\delta}(f^k|\nu^k) = k \cdot \Omega(\mathrm{IC}_{\mu,1/20,1/10,\delta/k}(f|\nu))$

We now give the slight generalization of the MWY theorem that is used to prove Theorem 1.6.

▶ **Theorem 4.2** (Slight generalization of the direct-sum theorem of [16])**.** *Let $X \in \mathcal{X}$, $Y \in \mathcal{Y}$ and λ be a distribution on (X,Y,D) with marginals μ over (X,Y) and ν over D such that*

for every value d of D, X and Y are conditionally independent given $D = d$. For any $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, $k \in \mathbb{N}$ and $\epsilon \leq 1/3$, $\mathrm{IC}_{\mu^k,\epsilon}(f^k|\nu^k) = k \cdot \Omega(\mathrm{IC}_{\mu,O(\epsilon),O(\epsilon/k)}(f|\nu))$.

Proof. For every $i \in [k]$, we denote by W_i the pair (X_i, Y_i) and by $f(W_{<i})$ the tuple $(f(W_1), \ldots, f(W_{i-1}))$.

▶ **Definition 1.3** (Good indices)**.** An index $i \in [k]$ is said to be *good* if

$$\Pr_{\mu,\Pi}[\Pi_i(W) = f(W_i)|\Pi_{<i}(W) = f(W_{<i})] = 1 - O(\epsilon/k)$$

▶ **Lemma 1.4.** *At least half of the indices $i \in [k]$ are good.*

Proof. Follows from averaging and the fact that

$$\Pr_{\mu,\Pi}[\Pi(W) = f(W)] = \prod_{i=1}^{k} \Pr_{\mu,\Pi}[\Pi_i(W) = f(W_i)|\Pi_{<i}(W) = f(W_{<i})] \geq 1 - \epsilon. \quad \blacktriangleleft$$

▶ **Definition 1.5** (Reasonable prefixes)**.** Fix a good index $i \in [k]$. A prefix $w_{<i}$ is said to be *reasonable* if

1. $I(\Pi(W); W|\nu^k, W_{<i} = w_{<i}) = O(I(\Pi(W); W|\nu^k))$
2. $\Pr_{\mu,\Pi}[\Pi_{<i}(W) = f(W_{<i})|W_{<i} = w_{<i}] = 1 - O(\epsilon)$
3. $\Pr_{\mu,\Pi}[\Pi_i(W) = f(W_i)|\Pi_{<i}(W) = f(W_{<i}), W_{<i} = w_{<i}] = 1 - O(\epsilon/k)$

▶ **Lemma 1.6.** *For every good index $i \in [k]$, a random prefix $w_{<i}$ is reasonable with probability at least $1/2$.*

Proof. Follows from 3 applications of Markov's inequality, the union bound and sufficiently large constants in the $O(\cdot)$ notations. ◀

▶ **Definition 1.7** (Acceptable fixings d_{-i})**.** Fix a good index $i \in [k]$ and a reasonable prefix $w_{<i}$. A fixing d_{-i} of D_{-i} is said to be *acceptable* if

1. $I(\Pi(W); W|\nu^k, W_{<i} = w_{<i}, D_{-i} = d_{-i}) = O(I(\Pi(W); W|\nu^k, W_{<i} = w_{<i}))$
2. $\Pr_{\mu,\Pi}[\Pi_{<i}(W) = f(W_{<i})|W_{<i} = w_{<i}, D_{-i} = d_{-i}] = 1 - O(\epsilon)$
3. $\Pr_{\mu,\Pi}[\Pi_i(W) = f(W_i)|\Pi_{<i}(W) = f(W_{<i}), W_{<i} = w_{<i}, D_{-i} = d_{-i}] = 1 - O(\epsilon/k)$

▶ **Lemma 1.8.** *Fix a good index $i \in [k]$ and a reasonable prefix $w_{<i}$. Then, a random fixing d_{-i} of D_{-i} is acceptable with probability at least $1/2$.*

Proof. Follows from 3 applications of Markov's inequality, the union bound and sufficiently large constants in the $O(\cdot)$ notations. ◀

▶ **Lemma 1.9.** *Fix a good index $i \in [k]$, a reasonable prefix $w_{<i}$ and an acceptable fixing d_{-i}. Then, we have that:*

$$I(\Pi(W); W|\nu^k, W_{<i} = w_{<i}, D_{-i} = d_{-i}) \geq \mathrm{IC}_{\mu,O(\epsilon),O(\epsilon/k)}(\mathrm{HAM}_{n,1}|\nu)$$

Proof. The new protocol Π' simulates the old protocol with $W_{<i} = w_{<i}$ and $D_{-i} = d_{-i}$ hardwired and it doesn't use any public randomness beyond that of the old protocol. Hence,

$$I(\Pi(W); W|\nu^k, W_{<i} = w_{<i}, D_{-i} = d_{-i}) \geq I(\Pi'(W_i); W_i|\nu). \quad \blacktriangleleft$$

The lemma now follows from the chain rule for mutual information and Lemmas 1.4, 1.6 and 1.8. ◀

An Approximate Version of the Tree Packing Conjecture via Random Embeddings*

Julia Böttcher[1], Jan Hladký[2], Diana Piguet[3], and Anusch Taraz[4]

1 **Department of Mathematics, London School of Economics**
Houghton Street, London, WC2A 2AE, UK
j.boettcher@lse.ac.uk

2 **DIMAP and Mathematics Institute, University of Warwick**
Coventry, CV4 7AL, UK
honzahladky@gmail.com

3 ***Current affiliation:*** **European Centre of Excellence, NTIS, Faculty of Applied Sciences, University of West Bohemia**
Univerzitní 22, 306 14, Pilsen, Czech Republic
Previous affiliation: **School of Mathematics, University of Birmingham, Edgbaston, Birmingham, B15 2TT, UK**
diana.piguet@gmail.com

4 **Institut für Mathematik, Technische Universität Hamburg-Harburg**
Schwarzenbergstrasse 95, Gebäude E, 21073 Hamburg, Germany
taraz@tuhh.de

Abstract

We prove that for any pair of constants $\varepsilon > 0$ and Δ and for n sufficiently large, every family of trees of orders at most n, maximum degrees at most Δ, and with at most $\binom{n}{2}$ edges in total, packs into $K_{(1+\varepsilon)n}$. This implies asymptotic versions of the well-known tree packing conjecture of Gyárfás from 1976 and another tree packing conjecture of Ringel from 1963 for trees with bounded maximum degree. A novel random tree embedding process combined with the nibble method forms the core of the proof.

1998 ACM Subject Classification G.2.2 Graph Theory, G.3 Probability and Statistics

Keywords and phrases tree packing conjecture, Ringel's conjecture, random walks, quasirandom graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.490

1 Introduction

Tree embeddings and packings, albeit their simple formulation, have proven to be among the most difficult tasks in graph theory. In 1963, Erdős and Sós conjectured that every graph with average degree larger than $k-1$ must contain a copy of every tree on $k+1$ vertices. In close vicinity to this problem, Loebl, Komlós and Sós conjectured in 1995 that the same holds when substituting the median degree for the average degree. A solution to the first conjecture has been announced by Ajtai, Komlós, Simonovits and Szemerédi in the early 1990s. In 2008, the dense case of the second conjecture has been proven to be true by

* The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. PIEF-GA-2009-253925. J. H. is an EPSRC Research Fellow. A. T. was supported in part by DFG grant TA 319/2-2.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 490–499

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

Hladký and Piguet [14] and Cooley [9], and an approximate version of the general case was confirmed recently by Hladký, Komlós, Piguet, Simonovits, Stein, and Szemerédi [13].

The focus of this paper is on packing trees, which generalises the notion of embeddings to finding several subgraphs simultaneously. A family of graphs $\mathcal{H} = (H_1, \dots, H_k)$ is said to *pack* into a graph G if there exist pairwise edge-disjoint copies of $H_1, \dots, H_k$ in G. In 1976, Gyárfás (who, according to his own words, was fascinated and motivated by the fact that $\sum_{i=1}^{n} i = n(n-1)/2$) proposed the following conjecture that is often referred to as the Tree Packing Conjecture.

▶ **Conjecture 1.** *Any family $(T_1, T_2, \dots, T_n)$ of trees, with $v(T_j) = j$ with $j \in [n]$, packs into K_n.*

A related conjecture of Ringel [18], dating back to 1963, deals with packing many copies of the same tree.

▶ **Conjecture 2.** *Any $2n+1$ identical copies of any tree of order $n+1$ pack into K_{2n+1}.*

Note that this conjecture, too, proposes the existence of a *perfect packing*, which means that the number of edges in the host graph equals to the total number of edges to be packed.

In this extended abstract, we outline a proof of a common generalisation that confirms the approximate correctness of Conjectures 1 and 2 for bounded-degree trees, without needing any further requirement than just the obvious upper bound on the total number of edges.

▶ **Theorem 3.** *For any $\varepsilon > 0$ and any $\Delta \in \mathbb{N}$, there is an $n_0 \in \mathbb{N}$ such that for any $n \geq n_0$ the following holds. Any family of trees $\mathcal{T} = (T_i)_{i\in[k]}$ such that T_i has maximum degree at most Δ and order at most n for each $i \in [k]$, and $\sum_{i\in[k]} e(T_i) \leq \binom{n}{2}$ packs into $K_{(1+\varepsilon)n}$.*

So far other major steps towards the resolution of these two conjectures have been comparatively limited. We briefly review them in the following (see also the slightly outdated survey by Hobbs [15]). Focussing on the initial set of smaller trees appearing in Conjecture 1, Bollobás [4] proved that any family of trees $T_1, \dots, T_s$ with $v(T_i) = i$ and $s < n/\sqrt{2}$ can be packed into K_n. Moreover, he observed that the validity of Erdős–Sós conjecture would imply that one can improve the bound to $s < \frac{1}{2}\sqrt{3}n$. Yuster [22] considered packings of trees into complete bipartite graphs and proved that any sequence of trees $T_1, \dots, T_s$, $s < \sqrt{5/8}n$ can be packed into $K_{n-1,n/2}$. This improves upon a result of Caro and Roditty [6] and is related to a conjecture of Hobbs, Bourgeois and Kasiraj [16]. Furthermore, a result of Caro and Yuster [7] implies the existence of a perfect packing of a family of trees into K_n, provided that the trees are very small compared to n.

In contrast, packing the larger trees that appear in Conjecture 1 has turned out to be a far more challenging task. Recently, Balogh and Palmer [2] proved that any family of trees $T_n, T_{n-1}, \dots, T_{n-\frac{1}{10}n^{1/4}}$, $v(T_i) = i$ packs into K_{n+1}.

In addition, special classes of tree families have been investigated. Progress so far mainly concerns trees that are similar to stars or paths. Already in the starting paper [12], Conjecture 1 is proven to hold in the special case when all the trees are stars and paths. Dobson [10] and Hobbs, Bourgeois and Kasiraj [16] consider packings of trees with small diameter.

Finally, we remark that it is known that the degree sequences of the trees appearing in Conjecture 1 can be matched up to fit into the complete graph: Fishburn [11] proved that if we fill up each tree T_i by adding $n-i$ isolated vertices and let $d_{i,1}, \dots, d_{i,n}$ denote the degree sequence of the resulting forest, then there are permutations $\pi_1, \dots, \pi_n$ such that $\sum_i d_{i,\pi_i(j)} = n-1$ for all $j \in [n]$.

2 Strategy and Preliminaries

In rough terms the basic idea for our proof of Theorem 3 is as follows. We use a random process to pack the trees. During this process, we keep checking that the remaining host graph (composed of the edges where no tree edge has been embedded yet) continues to satisfy certain quasirandom properties with high probability. The quasirandomness guarantees that we can carry on with our embedding as before.

In the following we will flesh out this agenda a bit further. In Section 3 we then recapitulate the main steps of the proof.

2.1 Quasirandomness

We start by recalling the concept of quasirandom graphs, which goes back to Thomason [21], and Chung, Graham, and Wilson [8].

▶ **Definition 4** (Quasirandom)**.** We say that a graph G of order n is α-*quasirandom of density* d if for every $B \subseteq V(G)$ we have $e(B) = d\binom{|B|}{2} \pm \alpha n^2$ edges.

A well-known feature of quasirandom graphs that is particularly important for our purposes is that we can control the size of the joint neighbourhood of almost all sets of vertices of size ℓ for constant ℓ.

▶ **Lemma 5.** *For every $\beta > 0$ and every integer $\Delta \geq 1$ there is a constant $\alpha > 0$ so that in every α-quasirandom graph $G = (V, E)$ of density $d \geq \beta$ for every given set $B \subseteq V$ and any $1 \leq \ell \leq \Delta$ all but at most $\beta\binom{n}{\ell}$ sets $\{v_1, \dots, v_\ell\} \subseteq V$ have a joint neighbourhood of size $(d^\ell \pm \beta)|B|$ in B.*

For the proof of Theorem 3 it is convenient to disregard vertices contained in too many sets that are exceptional in the sense of Lemma 5. This leads to the following definition.

▶ **Definition 6** (Superquasirandom)**.** We say that a graph $G = (V, E)$ is (α, Δ)-*superquasirandom* if for all $v \in V$, and for all $p \in [\Delta]$, we have

$$\left|\left\{S \in \tbinom{V}{p-1} \colon \mathrm{N}(S \cup \{v\}) \neq (1 \pm \alpha) d^p |V|\right\}\right| \leq \alpha \tbinom{|V|}{p-1} ,$$

where $\mathrm{N}(X)$ denotes the joint neighbourhood of vertices in the set X.

A consequence of Lemma 5 is that each quasirandom graph contains an almost spanning superquasirandom graph (where Δ is fixed and the parameter α is slightly worse than the original quasirandomness parameter).

2.2 Probabilistic Tools

For the analysis we shall use only two relatively standard bounds, McDiarmid's Inequality and Suen's Inequality, which we now introduce. Suppose that $\Omega = \prod_{i=1}^{k} \Omega_i$ is a product probability space. A measurable function $f : \Omega \to \mathbb{R}$ is said to be C-*Lipschitz* if for each $\omega_1 \in \Omega_1, \omega_2 \in \Omega_2, \dots, \omega_i, \omega_i' \in \Omega_i, \dots, \omega_k \in \Omega_k$ we have

$$|f(\omega_1, \omega_2, \dots, \omega_i, \dots, \omega_k) - f(\omega_1, \omega_2, \dots, \omega_i', \dots, \omega_k)| \leq C .$$

McDiarmid's Inequality (see [17]) states that Lipschitz functions are concentrated around their expectation.

▶ **Lemma 7** (McDiarmid's Inequality). *Let $f : \Omega \to \mathbb{R}$ be a C-Lipschitz function defined on a product probability space $\Omega = \prod_{i=1}^{k} \Omega_i$. Then for each $t > 0$ we have*

$$\mathbb{P}\big[|f - \mathbb{E}[f]| > t\big] \leq 2\exp\left(-\frac{2t^2}{C^2 k}\right) .$$

Next, we state Suen's inequality ([20], see also [1, p. 128]). Let $\{B_i \subseteq \Omega\}_{i\in I}$ be a finite collection of events in an arbitrary probability space Ω. A *superdependency graph* for $\{B_i\}_{i\in I}$ is an arbitrary graph on the vertex set I whose edges satisfy the following. Let $I_1, I_2 \subseteq I$ be any two disjoint sets with no edge crossing from I_1 to I_2. Then any Boolean combination of the events $\{B_i\}_{i\in I_1}$ is independent of any Boolean combination of the events $\{B_i\}_{i\in I_2}$. In this setting (and only in this setting) we write $i \sim j$ to denote that ij forms an edge. Suen's Inequality allows us to approximate $\mathbb{P}[\bigwedge \overline{B_i}]$ by $\prod \mathbb{P}[\overline{B_i}]$.

▶ **Lemma 8** (Suen's Inequality). *Using the above notation, we have*

$$\left|\mathbb{P}\Big[\bigwedge \overline{B_i}\Big] - \prod \mathbb{P}[\overline{B_i}]\right| \leq \prod \mathbb{P}[\overline{B_i}] \cdot \Big(\exp\big(\sum_{i\sim j} \nu_{i,j}\big) - 1\Big),$$

where $\nu_{i,j} = \frac{\mathbb{P}[B_i \wedge B_j] + \mathbb{P}[B_i]\mathbb{P}[B_j]}{\prod_{\ell \sim i \text{ or } \ell \sim j}(1-\mathbb{P}[B_\ell])}$.

2.3 Nibble Rounds

In this subsection we specify two natural ways to design a random embedding process for trees. The random embedding process we use in our proof requires some further variations, which we shall describe in the next subsection.

First, consider the following approach: successively build a packing h of the trees, edge by edge, starting with an arbitrary edge in an arbitrary tree and then following the structure of the trees. Here, when embedding an edge xy of some tree T_i, with x already embedded to $h(x)$, we choose a random neighbour $v \in V(G)$ of $h(x)$ that is not contained in the set $U_i \subseteq V(G)$ of T_i-images so far, and embed y to $h(y) := v$. After embedding xy, we remove the edge uv from G and add v to U_i. Clearly, this process produces a proper packing unless we get stuck, that is, unless the set $\mathrm{N}_G\big(h(x)\big) \setminus U_i$ gets empty. But if, during the evolution, the host graph G always remains sufficiently quasirandom, then with high probability $\mathrm{N}_G\big(h(x)\big) \setminus U_i$ should not get empty (because $e(K_{(1+\varepsilon)n}) - \sum_{i\in[k]} e(T_i) \geq \varepsilon n^2$ implies that G has positive density throughout).

It seems likely hat the host graph does indeed remain quasirandom in this process, but unfortunately graph processes like this one are rather difficult to analyse because of their dynamically evolving environment in each step. To circumvent these difficulties we have adopted a *nibble* approach by proceeding in constantly many rounds and updating the environment only after each round. This method was pioneered by Rödl [19] to prove the existence of asymptotically optimal Steiner systems (see [1] for an exposition). Since then it has served as an important ingredient for several breakthroughs in combinatorics. In our setting the nibble method could amount to the following approach for embedding $T_1, \dots, T_k$ into $G = K_{(1+\varepsilon)n}$:

- Pack the trees in r rounds (with r big but constant). For this purpose, cut each tree T_i into small equally sized forests F_i^j with $j \in [r]$. In round j embed exactly one forest of each tree T_i, i.e., the forests $F_1^j, F_2^j, \dots, F_k^j$.

- In round j, for each i construct a *random homomorphism* from the forest F_i^j to G as follows. First, randomly embed some forest vertex x, then choose a neighbour v uniformly at random in $\mathrm{N}_G(h(x)) \setminus U_i$, where the *forbidden set* $U_i \subseteq V(G)$ are vertices used by T_i in previous rounds. Then continue with the next vertex in F_i^j, following again the structure of T_i.
- After round j, delete all the edges from G to which some forest edges were mapped in this round and add to U_i all images of vertices of F_i^j.

In other words, the difference between this approach and the random process described in the beginning of this subsection, is that the host graph G and the sets U_i are not updated after the embedding of each single vertex, but only at the end of each round.

Obviously, this procedure may not produce a proper packing of the trees: Firstly, it could create *vertex collisions*, where two vertices of some tree T_i are mapped to the same vertex of the host graph G. Secondly, there could be *edge collisions*, where two edges of different trees are mapped to the same edge. But since all forests F_i^j are small, this will create only a small proportion of vertex and edge collisions in each round, and the vertex and edge deletions at the end of each round guarantee that there are no collisions between rounds. Because our host graph has $(1+\varepsilon)n$ vertices it turns out that these few collisions are easy to "repair" by reembedding vertices with the help of a simple greedy strategy (see also Section 3).

2.4 Dependencies

The difficulty with analysing the random homomorphisms described above (sometimes called *tree-indexed random walks*) is that dependencies between embedded vertices are difficult to control. Recently, Barber and Long developed techniques that allow to handle these dependencies. In particular, in [3] they show that the image of a bounded-degree tree of order $\Theta(n^2)$ in a dense quasirandom graph of order n using a random homomorphism is again a quasirandom graph with high probability. It seems likely that the techniques from [3] could be used to prove the similar but more complicated properties that form the core of our proof.

Our approach (which was developed before the techniques of Barber and Long) is different. We instead use the following construction of random homomorphisms, which we call *limping homomorphisms*, in round j of the nibble approach described above.

- For each i, call one of the colour classes of F_i^j the set P of *primary vertices*, and the other the set S of *secondary vertices*. First map all primary vertices randomly to vertices of $V(G) \setminus U_i$. Then map each secondary vertex randomly into the common $(G - U_i)$-neighbourhood of the images of its forest neighbours — unless the size of the common neighbourhood is not as expected, in which case we simply *skip* this secondary vertex.

One crucial observation is that Lemma 5 asserts that if G is quasirandom in each round, then few vertices are skipped. We shall argue that this is the case in Section 2.6.

For the arguments presented below, notice that a realization of the limping homomorphism can be represented by an element of the probability space

$$\Omega_F = V^P \times [0,1]^S \,. \tag{1}$$

Here, the $[0,1]^S$-component indicates the relative positions of the images of the secondary vertices in the list of the common neighbours of the images of the respective primary vertices.

In Section 2.5, we illustrate some basic properties of limping homomorphisms. Then, in Section 2.6 we give, as an illustration of our proof method, a short proof of the main result of [3] when tree-indexed random walks are replaced by limping homomorphisms. This result forms one main component of our proof in a simplified setting.

2.5 Limping Homomorphism on Quasirandom Graphs

In this section we show that limping homomorphisms on quasirandom graphs behave very much like tree-indexed random walks. In particular, vertices of the images are spread uniformly over the graph, and so are the edges. Moreover, there are very few skipped vertices.

▶ **Lemma 9.** *Suppose that we are given $\alpha \in (0, \frac{1}{4})$, a tree T of maximum degree at most Δ with a bipartition into primary and secondary vertices, and an (α, Δ)-superquasirandom graph $G = (V, E)$ of density d. Let h be the limping homomorphism from T to G. Let $u, v \in V$, let $x \in V(T)$ be an arbitrary primary vertex, let $y \in V(T)$ be an arbitrary secondary vertex. Then the following statements hold.*

(a) $\mathbb{P}[h(x) = v] = \frac{1}{|V|}$.

(b) $\mathbb{P}[y \text{ is skipped}] \leq \alpha$.

(c) $\mathbb{P}[h(y) = v] = \frac{\left(1 \pm \alpha\left(\frac{2}{d}\right)^{\Delta}\right)^{\Delta+3}}{|V|}$.

(d) *Suppose that $xy \in E(T)$ and $uv \in E$. Then* $\mathbb{P}[h(x) = u \text{ and } h(y) = v] = \frac{\left(1 \pm \alpha\left(\frac{2}{d}\right)^{\Delta}\right)^{\Delta+2}}{d|V|^2}$.

(e) $\mathbb{P}\left[\exists z \in V(T) \setminus \{x\} \,:\, h(x) = h(z)\right] \leq \frac{v(T)}{|V|}$.

(f) $\mathbb{P}\left[\exists z \in V(T) \setminus \{y\} \,:\, h(y) = h(z)\right] \leq \frac{2v(T)}{d^{\Delta}|V|}$.

(g) *For the number of colliding vertices* $\mathrm{VC} = \{z \in V(T) \colon \exists z' \,:\, h(z) = h(z')\}$ *and every* $t > 0$ *we have* $\mathbb{P}\left[|\mathrm{VC}| \geq \frac{2v(T)^2}{d^{\Delta}|V|} + t\right] \leq 2\exp(-\frac{t^2}{2(\Delta+1)^2 v(T)})$.

As an example of the methods used for obtaining this lemma, we include the proofs of 4 and 7.

Proof of Lemma 94 and 7.

4 Let A be the event that x gets mapped to u, let B be the event that y gets mapped to v, let C be the event that y is not skipped, and let D be the event that v is in the common neighbourhood of $h(\mathrm{N}_T(y) \setminus \{x\})$. Note that $B \subseteq C \cap D$. Let $\mathcal{E}_q$ be the event that $|h(\mathrm{N}_T(y))| = q + 1$. As D and A are independent even if we condition on $\mathcal{E}_q$, we have

$$\begin{aligned}\mathbb{P}\left[A \cap B | \mathcal{E}_q\right] &= \mathbb{P}\left[A \cap B \cap C \cap D | \mathcal{E}_q\right] \\ &= \mathbb{P}\left[A|\mathcal{E}_q\right] \cdot \mathbb{P}\left[D|A \cap \mathcal{E}_q\right] \cdot \mathbb{P}\left[C|\mathcal{E}_q \cap D \cap A\right] \cdot \mathbb{P}\left[B|\mathcal{E}_q \cap C \cap D \cap A\right] \\ &= \mathbb{P}\left[A|\mathcal{E}_q\right] \cdot \mathbb{P}\left[D|\mathcal{E}_q\right] \cdot \mathbb{P}\left[C|\mathcal{E}_q \cap D \cap A\right] \cdot \mathbb{P}\left[B|\mathcal{E}_q \cap C \cap D \cap A\right] \,. \qquad (2)\end{aligned}$$

We have $\mathbb{P}\left[A|\mathcal{E}_q\right] = \mathbb{P}\left[A\right] = \frac{1}{|V|}$. As G is (α, Δ)-superquasirandom, $\deg(v) = (1 \pm \alpha)d|V|$. Consequently, $\mathbb{P}\left[D|\mathcal{E}_q\right] = \big((1 \pm \alpha)d\big)^q$. The number of q-sets $\{v_1, \ldots, v_q\}$ in $\mathrm{N}(v)$ with $|\mathrm{N}(v_1, \ldots, v_q, u)| \neq (1 \pm \alpha)d^{q+1}|V|$ is at most $\alpha\binom{|V|}{q}$. As $|\mathrm{N}(v)| \geq (1 - \alpha)d|V|$, the total number of $(q+1)$-sets that contain u and have the remaining vertices in $\mathrm{N}(v)$ is at least $\binom{(1-\alpha)d|V|}{q}$. We thus get

$$1 \geq \mathbb{P}\left[C|\mathcal{E}_q \cap D \cap A\right] \geq 1 - \frac{\alpha\binom{|V|}{q}}{\binom{(1-\alpha)d|V|}{q}} \geq 1 - \alpha\left(\tfrac{2}{d}\right)^{\Delta} \,,$$

where we use $(1 - \alpha)d|V| - q \geq \frac{1}{2}d|V|$, which follows from $|V| \geq 4\Delta/d$. Finally, if y is not skipped, then $|N(h(\mathrm{N}_T(y)))| = (1 \pm \alpha)d^{q+1}|V|$, implying that

$$\mathbb{P}\left[B|\mathcal{E}_q \cap C \cap D \cap A\right] = \left((1 \pm \alpha)d^{q+1}|V|\right)^{-1} .$$

The claimed bound now follows by substituting the above estimates into (2).

7 Using the bounds from 5 and 6, we get $\mathbb{E}\left[|\mathrm{VC}|\right] \leq \frac{2v(T)^2}{d^\Delta |V|}$. To prove concentration, consider the product space Ω_T from (1) and view $|\mathrm{VC}|$ as a function τ from Ω_T to $\mathbb{R}$. We claim that $|\mathrm{VC}|$ is $2(\Delta+1)$-Lipschitz. Indeed, if the random real $\tau(y)$ changes for a single secondary vertex y, this only affects the embedding of y and hence changes $|\mathrm{VC}|$ by at most 2. If, on the other hand, for a single primary vertex x the random choice of $h(x)$ changes, then only the embedding of x and possibly its neighbours is affected. In this case $|\mathrm{VC}|$ changes by at most $2(\Delta+1)$. McDiarmid's Inequality (Lemma 7) implies then that

$$\mathbb{P}\left[|\mathrm{VC}| \geq \tfrac{2v(T)^2}{d^\Delta |V|} + t\right] \leq \mathbb{P}\Big[|\mathrm{VC}| \geq \mathbb{E}\big[|\mathrm{VC}|\big] + t\Big] \leq 2\exp\left(-\frac{2t^2}{(2(\Delta+1))^2 v(T)}\right). \qquad \blacktriangleleft$$

2.6 Images of Limping Homomorphisms are Quasirandom

To illustrate our techniques, we prove that the image of the limping homomorphism of a bounded-degree forest of order $\Theta(n^2)$ in a quasirandom graph of order n is typically again a quasirandom graph. Even though this statement directly does not appear in our proof of Theorem 3, some more involved variants of it do. We emphasise that the extremely short range of correlations in limping homomorphisms allow for a relatively easy proof, which is in particular much shorter than the proof of the corresponding statement for tree-indexed random walks verified in [3].

Suppose we are given an (α, Δ)-quasirandom graph G of density d on n vertices, for $0 < \alpha \ll \beta \ll d$. Let F be a forest of total size γn^2 whose degrees are bounded by a constant Δ, where $\gamma \in (0,1]$ is arbitrary. Let H be its image in G under the limping homomorphism. We prove that with high probability the graph H is a β-quasirandom graph.

Let α_1 and α_2 be such that $\alpha \ll \alpha_1 \ll \alpha_2 \ll \beta$. Suppose that $f \in E(F)$ and $e \in E(G)$. Let $A_{f,e}$ be the event that f is not mapped to e. An application of Lemma 94 gives that for each $f \in E(F)$ and most edges $e \in E(G)$ we have $\mathbb{P}[A_{f,e}] = 1 - (1 \pm \alpha')\frac{2}{dn^2}$. Now, fix such an edge e, and build an auxiliary superdependency graph on the vertex set $E(F)$ by connecting $f_1 \in E(F)$ to $f_2 \in E(F)$ if $\mathrm{dist}(f_1, f_2) \leq 2$. This is indeed a superdependency graph for the events $\left\{A_{f,e}\right\}_{f \in E(F)}$ with respect to the limping homomorphism. (This is the first point where we benefit from working with limping homomorphisms instead of random F-indexed walks.) Suen's Inequality gives us $\mathbb{P}[\wedge_f A_{f,e}] = (1 \pm \alpha_1)(1 - (1 \pm \alpha')\frac{2}{dn^2})^{e(F)} = (1 \pm \alpha_2)\exp(-\frac{2\gamma}{d})$. In particular, this quantity does not depend on the choice of the edge e.

Therefore, for each $B \subseteq V(G)$ we have $\mathbb{E}[|E(H) \cap \binom{B}{2}|] = d\binom{|B|}{2} \cdot (1 - \exp(-\frac{2\gamma}{d})) \pm \frac{\beta n^2}{2}$. Further, observe that the quantity $|E(H) \cap \binom{B}{2}|$ is Δ^2-Lipschitz. (Here again we make use of the short range of dependencies in limping walks.) Indeed, changing a position of a secondary vertex only changes the images of at most Δ edges incident to it. Changing a position of a primary vertex changes only the images of the edges at distance zero or one from that vertex. Thus, McDiarmid's Inequality gives

$$\mathbb{P}\left[\left|E(H) \cap \tbinom{B}{2}\right| \neq d\tbinom{|B|}{2} \cdot (1 - \exp(-\tfrac{2\gamma}{d})) \pm \beta n^2\right] \leq 2\exp\left(-\frac{2(\frac{\beta n}{2})^2}{\Delta^4 \cdot \gamma n^2}\right) = \exp(-\Theta(n^2)).$$

In particular, we can apply the union bound over all 2^n choices of the set B, and see that with high probability for each such set we have $\left|E(H) \cap \binom{B}{2}\right| = d\binom{|B|}{2} \cdot (1 - \exp(-\frac{2\gamma}{d})) \pm \beta n^2$. This proves the quasirandomness of H.

The above computation can be considered as a simplified version of a key argument in our proof, where we take G as the graph at the beginning of a nibble round j, and

$F = F_1^j \cup \ldots \cup F_k^j$. The simplification comes from the fact that we ignore the role of the forbidden sets U_i.

3 Outline of the Proof of Theorem 3

In this section we sketch how the ideas developed in Section 2 lead to a proof of Theorem 3. We follow the plan outlined in Section 2.3. That is, we cut the given bounded-degree trees $\{T_i\}_{i=1}^k$ into forests $\{F_i^1, \ldots, F_i^r\}_{i=1}^k$, where r is a fixed constant depending on ε and Δ only. We require for each $i \in [k]$ and $j \in [r]$ that the graph $F_i^1 \cup \ldots \cup F_i^j$ is a tree, and that $v(F_i^j) \approx \frac{v(F_i)}{r}$. This is possible because the trees T_i have bounded degrees.

We embed these forests into $K_{(1+\varepsilon)n}$ in r rounds. We start in round $j = 1$ with $G = K_{(1+\varepsilon)n}$. (Note that G is quasirandom.) In that round, we embed the forests $F_1^1, F_2^1, \ldots, F_k^1$ using limping homomorphisms. After the round, we update G by deleting the edges used by $F_1^1, F_2^1, \ldots, F_k^1$. Further, we create the forbidden sets $U_1, \ldots, U_k \subseteq V(K_{(1+\varepsilon)n})$ corresponding to the vertex images of $F_1^1, \ldots, F_k^1$. Using the techniques (simplified versions of which) we presented in Sections 2.5 and 2.6, we prove that, with high probability, G remains quasirandom (albeit with a worse parameter), the sets U_i are distributed in a random-like fashion over $V(K_{(1+\varepsilon)n})$, and the numbers of (vertex- and edge-) collisions and of skipped vertices are small. We then iterate this step in the next round. Throughout the whole embedding process, the graph G keeps getting sparser, but remains quasirandom and the forbidden sets U_i keep growing as further parts of the tree T_i are being added, but stay spread in a random-like way.

To take care of the vertex and edge collisions and of the skipped vertices, we set aside $\varepsilon n/2$ reserve vertices R of our original host graph $K_{(1+\varepsilon)n}$ before we actually start the embedding rounds described above. Throughout the nibble rounds, the limping homomorphisms avoid the set R. Then, at the end, a simple greedy strategy can be used to relocate vertices in collisions (and skipped vertices) to R, thus obtaining a proper packing. To make the greedy strategy work, we also need to guarantee that the collisions are well distributed over the host graph, implying that further invariants need to be controlled in the nibble rounds above.

4 Concluding Remarks

4.1 Strengthenings of Theorem 3

Theorem 3 does not hold for $\varepsilon = 0$. In [5] we construct an infinite sequence $\{\mathcal{T}_n\}_{n\in\mathcal{I}}$ of families of trees with maximum degree Δ, where $\mathcal{T}_n$ contains trees of orders n and has $\binom{n}{2}$ edges in total. Yet we show that $\mathcal{T}_n$ does not pack into K_n.

On the other hand, the following strengthening of Theorem 3 may be true: Any family of trees of orders at most n and maximum degrees at most Δ whose total number of edges is at most $\binom{n}{2}$ packs into K_{n+C_Δ}, for a suitable constant C_Δ depending on Δ only.

We are convinced that, at an expense of a more involved analysis, our techniques would allow to prove a version of Theorem 3 (for each fixed $\varepsilon > 0$) for Δ growing with n, possibly as big as $\Delta = O(\log^\alpha n)$ for some $\alpha > 0$.

Moreover, it could well be that Theorem 3 holds even for $\Delta = \frac{n}{2}$, but new techniques would be necessary for a proof. It can be shown (see [5]) that the family of $\ell := \left\lfloor \binom{n}{2} / ((\frac{1}{2} + 2\sqrt{\varepsilon})n) \right\rfloor$ copies of the star of order $(\frac{1}{2} + 2\sqrt{\varepsilon})n + 1$ does not pack in $K_{(1+\varepsilon)n}$. This shows that the $\frac{n}{2}$ barrier can essentially not be exceeded.

4.2 The Tree-packing Process

We expect that the random embedding process described in Section 2 performs well even as a dynamic process on an evolving graph. That is, we believe that the quasirandomness of the host graph is also maintained by a sequential random embedding of the trees, where we forbid the edges (globally) and vertices (just for that particular tree) immediately after they are used. This would yield another proof of Theorem 3, but we believe the analysis of this process would also be interesting in its own right.

4.3 Eliminating Dependencies

The key technical ingredient in our proof was to replace tree-indexed random walks by another process which behaves very similarly, but in which independence is regained extremely quickly. Such an approach may be useful elsewhere, in particular in the analysis of randomised algorithms.

Acknowledgements. JH wishes to thank Demetres Christofides, Gábor Kun and Oleg Pikhurko for helpful discussions.

Much of the work was done during research visits where we (JB, JH, DP) had to take our little children with us. We would like to acknowledge the support of the London Mathematical Society (JB, JH, DP), EPSRC Additional Sponsorship with grant reference EP/J501414/1 (DP), and the Mathematics Institute at the University of Warwick (JH) for contributing to childcare expenses that incurred during these trips.

The paper was finalised during the participation in the program *Graphs, Hypergraphs, and Computing* at Institut Mittag–Leffler. We would like to thank the organisers and the staff of the institute for creating a very productive atmosphere. Moreover, we would like to thank Emili Simonovits for helping us with babysitting.

References

1 N. Alon and J. H. Spencer. *The probabilistic method*. Wiley, Hoboken, NJ, third edition, 2008.

2 J. Balogh and C. Palmer. On the Tree Packing Conjecture. *SIAM J. Discrete Math.*, 27(4):1995–2006, 2013.

3 B. Barber and E. Long. Random walks on quasirandom graphs. *Electron. J. Combin.*, 20(4):Paper 25, 2013.

4 B. Bollobás. Some remarks on packing trees. *Discrete Math.*, 46(2):203–204, 1983.

5 J. Böttcher, J. Hladký, D. Piguet, and A. Taraz. An approximate version of the tree packing conjecture. arXiv:1404.0697.

6 Y. Caro and Y. Roditty. A note on packing trees into complete bipartite graphs and on Fishburn's conjecture. *Discrete Math.*, 82(3):323–326, 1990.

7 Y. Caro and R. Yuster. Packing graphs: the packing problem solved. *Electron. J. Combin.*, 4(1):Paper 1, 1997.

8 F. R. K. Chung, R. L. Graham, and R. M. Wilson. Quasi-random graphs. *Combinatorica*, 9(4):345–362, 1989.

9 O. Cooley. Proof of the Loebl-Komlós-Sós conjecture for large, dense graphs. *Discrete Math.*, 309(21):6190–6228, 2009.

10 E. Dobson. Packing trees into the complete graph. *Combin. Probab. Comput.*, 11(3):263–272, 2002.

11 P. C. Fishburn. Balanced integer arrays: a matrix packing theorem. *J. Combin. Theory Ser. A*, 34(1):98–101, 1983.

12 A. Gyárfás and J. Lehel. Packing trees of different order into K_n. In *Combinatorics (Proc. Fifth Hungarian Colloq., Keszthely, 1976)*, volume 18 of *Colloq. Math. Soc. János Bolyai*, pages 463–469. North-Holland, Amsterdam, 1978.

13 J. Hladký, J. Komlós, D. Piguet, M. Simonovits, M. Stein, and E. Szemerédi. The approximate Loebl–Komlós–Sós conjecture. arXiv:1211.3050.

14 J. Hladký and D. Piguet. Loebl–Komlós–Sós Conjecture: dense case. arXiv:0805:4834.

15 A. M. Hobbs. Packing trees. In *Proceedings of the Twelfth Southeastern Conference on Combinatorics, Graph Theory and Computing, Vol. II (Baton Rouge, La., 1981)*, volume 33, pages 63–73, 1981.

16 A. M. Hobbs, B. A. Bourgeois, and J. Kasiraj. Packing trees in complete graphs. *Discrete Math.*, 67(1):27–42, 1987.

17 C. McDiarmid. On the method of bounded differences. In *Surveys in combinatorics, 1989 (Norwich, 1989)*, volume 141 of *London Math. Soc. Lecture Note Ser.*, pages 148–188. Cambridge Univ. Press, Cambridge, 1989.

18 G. Ringel. Problem 25. In *Theory of Graphs and its Applications (Proc. Int. Symp. Smolenice 1963)*. Czech. Acad. Sci., Prague, 1963.

19 V. Rödl. On a packing and covering problem. *European J. Combin.*, 6(1):69–78, 1985.

20 W.-C. S. Suen. A correlation inequality and a Poisson limit theorem for nonoverlapping balanced subgraphs of a random graph. *Random Structures Algorithms*, 1(2):231–242, 1990.

21 A. Thomason. Pseudorandom graphs. In *Random graphs '85 (Poznań, 1985)*, volume 144 of *North-Holland Math. Stud.*, pages 307–331. North-Holland, Amsterdam, 1987.

22 R. Yuster. On packing trees into complete bipartite graphs. *Discrete Math.*, 163(1-3):325–327, 1997.

On Sharp Thresholds in Random Geometric Graphs

Milan Bradonjić[1] and Will Perkins[2]

1 **Bell Labs, Alcatel-Lucent**
600 Mountain Avenue 2C318, Murray Hill, NJ, USA
milan@research.bell-labs.com

2 **School of Mathematics, Georgia Tech**
686 Cherry St, Atlanta, GA, USA
perkins@math.gatech.edu

Abstract

We give a characterization of vertex-monotone properties with sharp thresholds in a Poisson random geometric graph or hypergraph. As an application we show that a geometric model of random k-SAT exhibits a sharp threshold for satisfiability.

1998 ACM Subject Classification G.3 Probability and Statistics

Keywords and phrases Sharp thresholds, random geometric graphs, k-SAT

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.500

1 Introduction

A property A of a discrete random structure is said to exhibit a *sharp threshold* with respect to a parameter p if there exists a $p_c = p_c(n)$ so that for every $\epsilon > 0$, for $p > (1+\epsilon)p_c$, A holds with probability $1 - o(1)$ and for $p < (1-\epsilon)p_c$, A holds with probability $o(1)$. The classic sharp thresholds in the Erdős-Rényi random graph $G(n,p)$ are the threshold for connectivity at $p = \log n/n$ and the threshold for a giant component at $p = 1/n$, see [2]. A property that does not exhibit a sharp threshold is that of containing a triangle: for any $c \in (0,\infty)$, when $p = c/n$ the probability that $G(n,p)$ contains a triangle is strictly bounded away from 0 and 1.

In addition to much investigation of the threshold location and behavior of specific properties of random graphs, a series of works have proved general threshold theorems. The first such result was by Bollobás and Thomason [6] showing that any monotone property A (a property closed under adding additional edges) has a threshold function: a $p^*(n)$ so that for $p \gg p^*$, $G(n,p)$ has property A with probability tending to 1, and for $p \ll p^*$, $G(n,p)$ has property A with probability tending to 0. Subsequently, Friedgut and Kalai [11] showed that every monotone property has a threshold width bounded by $O(\log^{-1} n)$: there is a function $C(\epsilon)$ so that for any $\epsilon > 0$, if $G(n,p)$ has property A with probability ϵ, then $G(n, p + C(\epsilon)/\log n)$ has property A with probability at least $1-\epsilon$. Bourgain and Kalai [7] improved this upper bound to $O(\log^{\delta-2} n)$ for any $\delta > 0$. Nevertheless, these theorems do not imply a sharp threshold in the sense defined above unless the critical probability for the property is sufficiently high.

Friedgut [10] gave a characterization of all monotone properties of random graphs that exhibit a sharp threshold: essentially they are properties that cannot be approximated by the property of containing a subgraph from a list of constant-size subgraphs. In other words, properties with coarse thresholds are all similar to the property of containing a triangle.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 500–514

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

Friedgut used his general theorem to prove that the satisfiability of a random k-SAT formula exhibits a sharp threshold, and then Achlioptas and Friedgut [1] used it to prove that the property of being k-colorable has a sharp threshold in $G(n,p)$. These properties had resisted previous analysis in part because of their complexity: determining the satisfiability of a k-SAT formula or the k-colorability of a graph are both NP-hard problems. In contrast, 2-SAT has a polynomial-time algorithm in the worst-case, and the threshold location [8] and width [5] of a random 2-SAT formula are both well understood.

In this paper we prove a general sharp threshold theorem for the *Random Geometric Graph* (RGG). The standard model of the RGG, $G_d(n,r)$, involves placing n points uniformly at random (or according to a Poisson process of intensity n) in $[0,1]^d$ (or the d-dimensional unit torus) and joining any two points at distance at most r by an edge. Unlike the edges in $G(n,p)$, the edges in the RGG are not independent. The RGG exhibits thresholds for some of the same properties as the Erdős-Rényi random graph. There is a unique giant component whose appearance occurs sharply at the threshold radius $r = \lambda_c n^{-1/d}$ [19]. The exact value of the constant λ_c is not known, but numerical simulations for $d = 2$ indicate $\lambda_c \approx 1.44$ [21] and bounds are given in [16, 14]. The RGG also has a sharp threshold for connectivity at $r = (\log n/(nV_d))^{1/d}$ [13, 18] (in the d-dimensional torus) where V_d is the volume of a unit ball in $\mathbb{R}^d$.

The RGG has been extensively studied in fields such as cluster analysis, statistical physics, hypothesis testing, and wireless sensor networks. One further application of the RGG is modeling data in a high-dimensional space, where the coordinates of the nodes of the RGG represent the attributes of the data. The metric imposed by the RGG then depicts the similarity between data elements in the high-dimensional space. See [4] or [19] for a survey of results on the RGG.

In the RGG, Goel et al. have shown that every monotone property has a threshold width (in terms of r) of $O(\log^{3/4} n/\sqrt{n})$ (for $d = 2$) and $O(\log^{1/d} n/n^{1/d})$ (for $d \geq 3$) [12]. This implies a sharp threshold in the sense described above when the critical radius of a property is sufficiently large, but not for sparser graphs, and in particular not in the connectivity or giant component regimes. For one-dimensional RGG's, McColm proved that every monotone property has a threshold function [15], in the sense of Bollobás-Thomason.

We prove a general criteria for sharp thresholds in the Poisson RGG. As an application, we introduce a geometric model of random k-SAT in which literals are placed at random in $[0,1]^d$, and prove that satisfiability exhibits a sharp threshold in this model. We also identify the location of this threshold in the case $k = 2$. Previously, a model of random k-SAT for $k = 1, 2$ with literals placed on a 2-dimensional lattice was proposed in [20], and in [17] the authors investigate a model of random k-XOR-SAT with finite interaction range, a kind of one-dimensional geometry.

The organization and main contributions of this paper are as follows:

1. In Section 2, we introduce notation, define two models of RGG's, and define a sharp threshold in each model. We then define analogous models of random geometric k-SAT.
2. In Section 3 we state our main result: a characterization of vertex-monotone properties with sharp thresholds in the Poisson RGG. We also state a result on transferring sharp thresholds from the Poisson to fixed-n model.
3. In Section 4 we state our results on random geometric k-SAT: for all $k \geq 2$, the satisfiability phase transition is sharp in the Poisson model. For $k = 2$, we find the location of this threshold.
4. Section 5 contains the proofs of the sharp threshold lemma and the sharpness of the satisfiability phase transition.
5. Sections 6–8 contain auxiliary results and proofs.

2 Models and Notation

We will denote point sets in $[0,1]^d$ by S, T and the graphs, hypergraphs or formulae formed by joining 2 (or k) points that appear in a ball of diameter r by G_S, G_T, F_S, F_T respectively.

We denote (hyper)graph properties by A and write $G \in A$ if graph G has property A. We say a property A holds 'with high probability' or 'whp' if $\Pr[G \in A] = 1 - o(1)$ as $n \to \infty$. We write $f(n) \sim g(n)$ if $f(n) = g(n)(1 + o(1))$.

We work with two models of random geometric graphs. For $n, d \in \mathbb{N}$ and $\mu, r \in \mathbb{R}^+$, $G_d(n, \mu, r)$ is the random graph formed by drawing a point set S according to a Poisson point process of intensity $n \cdot \mu$ on $[0,1]^d$ and then forming G_S by joining any two points at distance at most r. For the hypergraph version of this model, we form a k-uniform hyperedge on any set of k points in S that appear in a ball of diameter r. If $t > k$ points all appear in one ball of diameter r, then all $\binom{t}{k}$ possible k-uniform hyperedges are formed. The second model, $G_d(n, r)$, is the random graph drawn by placing n points uniformly and independently at random in $[0,1]^d$ to form S, then forming G_S by connecting points at distance at most r. Note that $G_d(n, r)$ has the same distribution as $G_d(n, \mu, r)$ conditioned on $|S| = n$. We use balls of diameter r instead of radius r to form the hypergraphs so as to match the definition of the RGG in the case $k = 2$.

We say a property A has *sharp threshold* in $G_d(n, \mu, r)$ if there exists a function $\mu^*(n), r(n)$ so that for any $\epsilon > 0$,

1. For $\mu > (1 + \epsilon)\mu^*$, $\Pr[G_d(n, \mu, r) \in A] = 1 - o(1)$.
2. For $\mu < (1 - \epsilon)\mu^*$, $\Pr[G_d(n, \mu, r) \in A] = o(1)$.

For $G_d(n, r)$ it is more convenient to describe a sharp threshold in terms of the probability that two random points in $[0,1]^d$ form an edge[1]. We write $r(p)$ for the radius that achieves edge probability p. With this definition, we say that a property A has *sharp threshold* in $G_d(n, r)$ if there exists a function $p^*(n)$ so that for any $\epsilon > 0$,

1. For $p > (1 + \epsilon)p^*$, $\Pr[G_d(n, r(p)) \in A] = 1 - o(1)$.
2. For $p < (1 - \epsilon)p^*$, $\Pr[G_d(n, r(p)) \in A] = o(1)$.

For the k-SAT problem, we will work with formulae on n boolean variables $x_1, \ldots, x_n$. A *literal* is a variable x_i or its negation $\overline{x}_i$. We say a formula $F \in SAT$ if F is satisfiable.

We define two random geometric distributions over k-SAT formulae, $F_k(n, \gamma)$ and $F_k(n, \mu)$:

1. $F_k(n, \gamma)$: Randomly place $2n$ points uniformly and independently in $[0,1]^d$ each labeled with the name of a unique literal in $\{x_1, \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n\}$. For any set of k literals that appear in a ball of diameter $r = \gamma n^{-1/d}$, form the corresponding k-clause and add it to the random formula.
2. $F_k(n, \mu)$: Draw independent Poisson point processes of intensity μ on $[0,1]^d$ for each of the $2n$ literals. For any set of k literals that appear in a ball of diameter $r = n^{-1/d}$, add the corresponding clause.

Note that $F_k(n, \gamma)$ with $\gamma = 1$ has the same distribution as $F_k(n, \mu)$ conditioned on each literal appearing exactly once.

In this work, we will consider k, γ, μ and d fixed with respect to n, and take asymptotics as $n \to \infty$. We use ℓ_∞ balls for simplicity in what follows, but all results hold for Euclidean balls as well, with constants involving the volume of the d-dimensional unit sphere.

[1] For constant dimension d, this definition is equivalent to asking for a critical threshold radius r^*, but for $d = d(n) \to \infty$, allowing r to increase by a factor $(1 + \epsilon)$ will cause a super-constant factor increase in the number of edges of the graph.

Another natural model to consider would be the following, call it $\tilde{F}(n,r)$: randomly place n points uniformly and independently in $[0,1]^d$, each labeled with the name of a variable $x_1,\dots x_n$ (instead of the name of a literal). Then for each set of k variables appearing in a ball of diameter r, add a k-clause with the signs of the k variables chosen uniformly and independently from the 2^k possible choices. The threshold behavior of satisfiability in $\tilde{F}(n,r)$ is simpler than in the other two models: the threshold is coarse, and determined locally by large cliques of variables (see Section 7).

3 Sharp Thresholds in Random Geometric Graphs

The following theorem characterizes vertex-monotone properties with sharp thresholds in the Poissonized random geometric graph $G_d(n,\mu,r)$. It is an application of Bourgain's theorem in the appendix of Friedgut's paper on sharp thresholds in random graphs [10].

▶ **Theorem 1.** *Let A be a vertex-monotone property of a k-uniform hypergraph that does not have a sharp threshold in $G_d(n,\mu,r)$. Then there exists constants $\epsilon,\delta,K>0$ independent of n so that for arbitrarily large n there is an $\alpha\in(\delta,1-\delta)$ so that either*

1. $\Pr_{G_d(n,\mu,r)}[\exists H\subseteq S:|H|\le K, G_H\in A]\ge\epsilon$,
   or
2. *There exists a point set T in $[0,1]^d$ with $|T|\le K$, $G_T\notin A$ so that*

$$\Pr[G_d(n,\mu,r)\in A|T\subseteq S]\ge\alpha+\epsilon\,.$$

with μ chosen so that $\Pr_{G_d(n,\mu,r)}[A]=\alpha$.

In other words, if a property does not have a sharp threshold, then either there is a constant probability that a constant-size witness of A exists in the RGG or there is a point set of constant size in $[0,1]^d$ that by itself does not have property A, but by conditioning on the presence of these points significantly raises the probability of A in the RGG. To prove that a property has a sharp threshold, we rule out both of these possibilities.

We can connect sharp thresholds in $G_d(n,\mu,r)$ with those in $G_d(n,r)$. In particular, if the threshold intensity $\mu^*(n)$ has a limit, then there is a sharp threshold edge probability p^* in $G_d(n,r)$, and it too is uniform in n, up to a technical condition on the form of the threshold density.

▶ **Proposition 2.** *Let $q(n)=a\log^b(n)n^{-c}$ be a decreasing function of n for constants a,b,c. Suppose a vertex and edge-monotone property A has a uniform sharp threshold in $G_d(n,\mu,r)$: there exists a constant μ^*, independent of n, so that*

1. *For $\mu>(1+\epsilon)\mu^*$, $\Pr[G_d(n,\mu,r(q(n)))\in A]=1-o(1)$*
2. *For $\mu<(1-\epsilon)\mu^*$, $\Pr[G_d(n,\mu,r(q(n)))\in A]=o(1)$,*

then A has a sharp threshold in $G_d(n,r)$ in a uniform sense: there exists a t^*, independent of n, so that

1. *For $p>(1+\epsilon)t^*q(n)$, $\Pr[G_d(n,r(p))\in A]=1-o(1)$.*
2. *For $p<(1-\epsilon)t^*q(n)$, $\Pr[G_d(n,r(p))\in A]=o(1)$.*

Here we think of $q(n)$ as a typical threshold function, for example: $1/n, c/n^2, \log^2 n/n$, etc. The technical condition on q is required to rule out properties whose definition depends non-uniformly on n, eg. for small n, A is the property of containing a triangle, while for large n, it is the property of containing an edge.

We conjecture that in fact all edge-monotone properties in $G_d(n,r)$ can be characterized similarly:

▶ **Conjecture 3.** *For every edge-monotone property A with a coarse threshold in $G_d(n, r(p))$ with respect to p, there are constants $K, \epsilon, \delta > 0$ so that for large n, $\alpha \in (\delta, 1-\delta)$, and p chosen so that $\Pr[G_d(n, r(p)) \in A] = \alpha$, either*

1. $\Pr_{G_d(n,r(p))}[\exists H \subseteq S : |H| \leq K, G_H \in A] \geq \epsilon$,
   or
2. *There exists a point set T in $[0,1]^d$ with $|T| \leq K$, $G_T \notin A$ so that*

$$\Pr[G_d(n, r(p)) \in A | T \subseteq S] \geq \alpha + \epsilon\,.$$

4 Random Geometric k-SAT

As an application of Theorem 1, we prove that in the $F_k(n, \mu)$ model, the threshold for satisfiability is sharp:

▶ **Theorem 4.** *For all k, there exists a function $\mu_k^*(n)$ so that for every $\epsilon > 0$,*

1. *For $\mu < \mu_k^*(n) - \epsilon$, $F_k(n, \mu) \in SAT$ whp.*
2. *For $\mu > \mu_k^*(n) + \epsilon$, $F_k(n, \mu) \notin SAT$ whp.*

Next, for $k = 2$ we determine the exact location of the satisfiability threshold in both models:

▶ **Theorem 5.** *For any $\epsilon > 0$,*

1. *If $\gamma < 2^{-(1+1/d)} - \epsilon$, then whp $F_2(n, \gamma) \in SAT$. If $\gamma > 2^{-(1+1/d)} + \epsilon$, then whp $F_2(n, \gamma) \notin SAT$.*
2. *If $\mu < 2^{-(d+1)/2} - \epsilon$, then whp $F_2(n, \mu) \in SAT$. If $\mu > 2^{-(d+1)/2} + \epsilon$, then whp $F_2(n, \mu) \notin SAT$.*

Note that from Proposition 8 in Section 6, both thresholds occur at $m = n$ clauses, matching the threshold for random 2-SAT. The proof of Theorem 5 is omitted in this extended abstract, but appears in the full version of the paper.

5 Proofs

5.1 Proof of Theorem 1

To prove Theorem 1, we discretize $[0,1]^d$ and place points independently at each gridpoint with a given probability. We apply Bourgain's theorem in a dual fashion, to the product space over positioned points instead of the product space of edges as in $G(n, p)$. We then show that with a fine enough discretization, the graph formed in the discrete model is identical to the graph formed in the Poisson model with high probability.

We will prove the theorem for labeled k-uniform hypergraphs, where the label set is $\{1, 2, \ldots, L(n)\}$ and the dimension $d = d(n)$ may be constant or tend to infinity with n. Points with label i will appear in $[0,1]^d$ according to a Poisson point process of intensity $n\mu/L$, with all labels appearing independently (thus the union of all labeled points is itself a Poisson point process of intensity $n\mu$). For a random geometric graph we can specialize to $k = 2$ with a single label. For random geometric k-SAT, the label set will have size $2n$, one label for each literal.

Place N^d grid points onto $[0,1]^d$ where $N = 16^d n^3$ so that gridpoint $(i_1, \ldots, i_d)$ is located at $((i_1 - 1/2)/N, \ldots (i_d - 1/2)/N)$ and each i_j ranges over $\{1, \ldots N\}$. To that gridpoint, assign the region $A_{i_1,\ldots,i_d} = ((i_1 - 1)/N, i/N] \times \cdots \times ((i_d - 1)/N, i_d/N]$. At each grid point, let each of the L possible labels appear independently with probability $p = \mu n / L N^d$ (more

than one label can appear at a single grid point). For every set of k labeled points that appear in a ball of diameter r (in l_2 or l_∞ distance, depending on the model), include the corresponding hyperedge in the hypergraph. The following proposition allows us to transfer results from the discrete model to the continuous model:

▶ **Proposition 6.** *There is a coupling of the discrete and continuous model so that with probability* $1 - o(1)$*, the labeled hypergraph generated by each is identical.*

Proof. We couple as follows: If at least one point with label l falls in the region $A_{i_1,\ldots,i_d}$ in the continuous model, let the label l be present on gridpoint $(i_1, \ldots, i_d)$ in the discrete model. If no point with label l falls in $A_{i_1,\ldots,i_d}$ in the continuous model, then flip an independent coin that is heads with probability

$$e^{\mu n/LN^d} \cdot (\mu n/LN^d - (1 - e^{-\mu n/LN^d})).$$

If the coin is heads, let l be present at $(i_1, \ldots, i_d)$.

The following facts suffice to prove the proposition:

- The coupling is faithful: the probability that gridpoint (i, j) has a point with label l is:

$$1 - e^{-\mu n/LN^d} + e^{-\mu n/LN^d} \cdot e^{\mu n/LN^d} \cdot (\mu n/LN^d - (1 - e^{-\mu n/LN^d})) = \mu n/LN^d$$

  and all gridpoints and literals are independent by construction.
- With probability $1 - o(1)$ no coins come up heads: i.e. no extra labeled points appear in the discrete model. The probability of heads for a single coin is $O((\mu n/LN^d)^2)$, and there are at most LN^d coins flipped. By the union bound whp no heads are flipped.
- With probability $1 - o(1)$ no two copies of any one label appear in the same $A_{i_1,\ldots,i_d}$. The probability that label l appears at least twice in a fixed $A_{i_1,\ldots,i_d}$ is $O((\mu n/LN^d)^2)$. There are N^d such boxes and L labels, so again whp no region contains more than one.
- With probability $1-o(1)$ no hyperedges disappear and no new hyperedges appear, moving from the continuous to the discrete model. In the coupling a point moves by at most $1/2N$ in each coordinate. For l_1, l_2, l_∞ norms this means the point moves at most $d/2N$ with respect to the norm. For a hyperedge to appear or disappear due to this movement, two points would need to begin at some distance $x \in [r - d/N, r + d/N]$. For a given pair of points uniformly distributed in $[0, 1]^d$, this occurs with probability that depends on the norm, but is bounded by $4^{d+1}dr/N$. Since the total number of points has a Poisson$(n\mu)$ distribution, we can condition, and whp have at most $2n\mu$ points. Taking the union bound over $\Theta(n^2)$ pairs of points gives a failure probability of $O(n^2 4^{d+1} dr/N) = o(1)$, from our choice of N and using the fact that $r \leq d$ and $d^2 \leq 4^d$.

◀

To complete the proof of Theorem 1, we apply the following theorem from Bourgain's appendix to Friedgut's work [10][2]. Bourgain's theorem gives a criteria for a monotone property on a product measure over the Hamming cube to have a sharp threshold, as opposed to Friedgut's result which applies only to random graphs and hypergraphs.

Consider a random subset $S \subseteq [N]$ with $i \in S$ with probability p, independently for all $1 \leq i \leq N$. Let A be a monotone property of subsets of $[N]$. (In the case of the random graph $G(n, p)$, $N = \binom{n}{2}$ and S is the set of present edges, A might be the property of having a triangle or connectedness.)

[2] For a recent explication of Bourgain's proof, see [3].

▶ **Theorem** (Bourgain [10]). *Assume that* $\Pr_p[A] = \alpha \in (0,1)$, $p \cdot d\Pr_p(A)/dp \leq C$ *and* $p = o(1)$. *Then there exists* $\delta(C,\alpha) > 0$ *so that either*

1. *the probability that* S *contains a subset* H *of constant size with* $H \in A$ *is greater than* δ, *or*
2. *there exists a constant-sized subset (e.g. a subgraph in* $G(n,p)$*)* $H \notin A$ *so that* $\Pr_p[Q|H \subseteq S] > \alpha + \delta$.

In other words, conditioning on the appearance of this constant-sized subset increases the probability of the property significantly. We apply this theorem directly to the discrete model above, with the product space $\{0,1\}^{LN^d}$ and $p = \mu n/LN^d$. A vertex-monotone property on random geometric graphs becomes a monotone property in this hypercube. Bourgain's theorem is applied as follows: if a property A does not have a sharp threshold, then by the mean value theorem there must be some μ so that $\Pr_\mu(A)$ is bounded away from 0 and 1, and $\mu \cdot d\Pr_\mu(A)/d\mu \leq C$, for some constant C. Then Bourgain's theorem asserts that either condition (1) or (2) must hold. The two conditions are equivalent in the discrete and continuous model since the graphs generated are identical with probability $1 - o(1)$.

5.2 Proof of Proposition 2

Let $t^* = (\mu^*)^c$. Fix $\epsilon > 0$.

First assume $p > (1+\epsilon)t^*q(n)$, and let $N = \frac{1}{\mu^*}(1+\epsilon/2)^{-c}n$. The conditions of Proposition 2 say that $\Pr[G_d(N, \mu^*(1+\epsilon/2)^{c/2}, r(q(N)) \in A] = 1 - o(1)$. From the concentration of a Poisson, with probability $1-o(1)$, the number of points drawn in $G_d(N, \mu^*(1+\epsilon/2)^{c/2}, r(q(N)))$ is bounded above by n. We also have

$$\begin{aligned} p &> (1+\epsilon)(\mu^*)^c q(n) \\ &= (1+\epsilon)(\mu^*)^c \frac{a\log^b n}{n^c} \\ &\geq \frac{a(1+\epsilon/2)\log^b(n/(\mu^*(1+\epsilon/2)^{-c}))}{(n/\mu^*)^c} \\ &= q(N) \end{aligned}$$

Since A is both vertex monotone and edge monotone, we have $\Pr[G_d(n, r(p)) \in A] = 1 - o(1)$.

Next assume $p < (1-\epsilon)t^*q(n)$, and let $N = \frac{1}{\mu^*}(1-\epsilon/2)^{-c}n$. The conditions say that $\Pr[G_d(N, \mu^*(1-\epsilon/2)^{c/2}, r(q(N)) \in A] = o(1)$. With probability $1 - o(1)$, the number of points drawn in $G_d(N, \mu^*(1-\epsilon/2)^{c/2}, r(q(N))$ is bounded below by n, and

$$\begin{aligned} p &< (1-\epsilon)(\mu^*)^c q(n) \\ &= (1-\epsilon)(\mu^*)^c \frac{a\log^b n}{n^c} \\ &\leq \frac{a(1-\epsilon/2)\log^b(n/(\mu^*(1-\epsilon/2)^{-c}))}{(n/\mu^*)^c} \\ &= q(N) \end{aligned}$$

And again since A is both vertex monotone and edge monotone, $\Pr[G_d(n, r(p)) \in A] = o(1)$.

5.3 k-SAT proofs

Proof of Theorem 4

To prove Theorem 4, we will assume that the threshold is coarse: i.e., there is some $\alpha \in (0,1)$ so that $\Pr_\mu(\text{UNSAT}) = \alpha$, for which $\mu \cdot d\Pr_\mu(\text{UNSAT})/d\mu \leq C$. It then suffices to rule out both possibilities in Theorem 1 to derive a contradiction. We will show: (1) whp there is no constant-sized set of positioned literals that is by itself unsatisfiable and (2) there is no constant-sized satisfiable 'booster', one that boosts the unsatisfiability probability from α to $\alpha + \epsilon$ when conditioned on. Using Proposition 10 (Section 8) we can assume that μ is a constant bounded from above and away from 0 independent of n.

Notation

We will denote by F_H the k-SAT formula generated by a set of positioned literals $H \subset [0,1]^d$. Let $G_\mu \subset [0,1]^d$ be a random set of positioned literals chosen according to $2n$ independent Poisson processes of intensity μ, one for each of the $2n$ literals: i.e. $F_k(n,\mu)$ has the distribution F_{G_μ}. We will use l_∞ distance to simplify calculations, but everything holds for l_2 or l_1 distance as well, with α_d, the volume of the d-dimensional unit ball replacing 2^d in the calculations below.

Condition 1: For any constant R, we show that whp there is no set of R positioned literals that form an unsatisfiable formula. We will use the *implication graph* of a 2-SAT formula: the directed graph on $2n$ vertices, each representing a literal in the formula, in which $l_1 \to l_2$ if the clause $(l_2 \vee \bar{l}_1)$ is in the formula. A *bicycle* (see eg. [8, 9]) of length L in a 2-SAT formula is a sequence of clauses

$$(u, w_1), (\overline{w}_1, w_2), (\overline{w}_2, w_3), \ldots, (\overline{w}_L, v)$$

where the w_i's are literals of distinct variables and $u, v \in \{w_1, \ldots, w_L\} \cup \{\overline{w}_1, \ldots, \overline{w}_L\}$. A 2-SAT formula is satisfiable if it does not contain a bicycle. Let Y_L be the number of bicycles of length L in F_{G_μ}. Then

$$\mathbb{E}Y_L \leq n^L 2^L (2L)^2 \Pr\left[(\overline{u}, w_1), (w_L, v) \in F_{G_\mu} \wedge \bigwedge_{i=1}^{L-1} (\overline{w}_i, w_{i+1}) \in F_{G_\mu}\right]. \tag{1}$$

▶ **Claim 7.** *The probability that a specified bicycle of length L appears in F_{G_μ} satisfies:*

$$\Pr\left[(\overline{u}, w_1), (w_L, v) \in F_{G_\mu} \wedge \bigwedge_{i=1}^{L-1} (\overline{w}_i, w_{i+1}) \in F_{G_\mu}\right] \leq \frac{\mu^2 + 3\mu + 1}{\mu^2} \left(\frac{2^d \mu^2}{n}\right)^{L+1},$$

where w_i's are literals of distinct variables and $u, v \in \{w_1, \ldots, w_L\} \cup \{\overline{w}_1, \ldots, \overline{w}_L\}$.

Proof. The literals in the above event are not all distinct, and so the clauses are not all independent. There may be two literals that are repeated as $\overline{u}$ and v, and perhaps $\overline{u} = v$. We consider three different cases for the overlapping clauses:

Case 1: $u \neq v$, $(u, v) \neq (\overline{w}_i, w_{i+1})$ for any i.

Say $u = w_i$ and $v = w_j$, though the argument will be the same if either or both is a negation. For $k \neq i-1$ or $j-1$, the clauses $(\overline{w}_k, w_{k+1})$ are independent of all other clauses in the bicycle. Each has probability of appearing $\sim 2^d\mu^2/n$ for our choice of μ. Now consider the pairs of clauses $\{(u = w_i, w_1), (\overline{w}_{i-1}, w_i)\}$ and $\{(\overline{w}_L, v = w_j), (\overline{w}_{j-1}, w_j)\}$.

The clauses within each pair are not independent, but the pairs are independent of each other. Both pairs are of the form $(l_1, l_2), (l_1, l_3)$ for distinct literals l_1, l_2, l_3. Conditioning on the number of appearances of l_1, we have

$$\begin{aligned}\Pr[(l_1,l_2),(l_1,l_3) \in F] &\sim \sum_{j=0}^{\infty} \frac{e^{-\mu}\mu^j}{j!} \left(1 - e^{-2^d \mu j/n}\right)^2 \\ &\sim \sum_{j=0}^{\infty} \frac{e^{-\mu}\mu^j}{j!} \frac{2^{2d}\mu^2 j^2}{n^2} \\ &= \frac{2^{2d}\mu^2}{n^2}\left(\mu + \mu^2\right) . \end{aligned} \tag{2}$$

All together, with the $L-3$ independent clauses, this gives that a bicycle of this type appears with probability at most

$$\left(\frac{2^{2d}\mu^3(\mu+1)}{n^2}\right)^2 \left(\frac{2^d\mu^2}{n}\right)^{L-3} = \frac{(\mu+1)^2}{\mu^2}\left(\frac{2^d\mu^2}{n}\right)^{L+1} .$$

Case 2: $u \neq v$, $(u, v) = (\overline{w}_i, w_{i+1})$ for some i.

For $k \neq i$, the clauses $(\overline{w}_k, w_{k+1})$ are independent of the other clauses in the bicycle. What remains is the triple $\{(u = \overline{w}_i, w_1), (\overline{w}_i, w_{i+1}), (w_L, w_{i+1})\}$. (The argument is the same if $u = w_{i+1}$ and $v = \overline{w}_i$). This triple is of the form $(l_1, l_2), (l_1, l_3), (l_4, l_3)$. We calculate the probability such a triple appears by conditioning on the number of appearances of l_1 and l_3:

$$\Pr[(l_1,l_2),(l_1,l_3),(l_4,l_3) \in F] \sim \sum_{j=0}^{\infty}\sum_{k=0}^{\infty} \frac{e^{-\mu}\mu^j}{j!}\frac{e^{-\mu}\mu^k}{k!}\frac{2^{3d}j^2k^2\mu^2}{n^3}$$

and so

$$\begin{aligned}\Pr[(l_1,l_2),(l_1,l_3),(l_4,l_3) \in F] &\sim \frac{2^{3d}\mu^2}{n^3}\sum_{j=0}^{\infty}\sum_{k=0}^{\infty} \frac{e^{-\mu}\mu^j}{j!}\frac{e^{-\mu}\mu^k}{k!} j^2k^2 \\ &= \frac{2^{3d}\mu^2}{n^3}(\mu+\mu^2)^2 \\ &= \frac{2^{3d}\mu^4(\mu+1)^2}{n^3} .\end{aligned}$$

Again all together the probability of the particular bicycle appearing is at most

$$\frac{2^{3d}\mu^4(\mu+1)^2}{n^3}\left(\frac{2^d\mu^2}{n}\right)^{L-2} = \frac{(\mu+1)^2}{\mu^2}\left(\frac{2^d\mu^2}{n}\right)^{L+1} .$$

Case 3: $u = v$.

Say $u = v = w_i$. (The same will work for $u = v = \overline{w}_i$). The clauses $(\overline{w}_k, w_{k+1})$ for $k \neq i-1$ are again independent of all other clauses in the bicycle. What remains are the clauses $(u = w_i, w_1), (\overline{w}_{i-1}, w_i), (w_L, v = w_i)$. This is a triple of the form $(l_1, l_2), (l_1, l_3), (l_1, l_4)$ and we calculate its probability by conditioning on the number of appearances of l_1:

$$\begin{aligned}\Pr[(l_1,l_2),(l_1,l_3),(l_1,l_4) \in F] &\sim \sum_{j=0}^{\infty}\frac{e^{-\mu}\mu^j}{j!}\frac{2^{3d}\mu^3 j^3}{n^3} \\ &= \frac{2^{3d}\mu^3}{n^3}(\mu^3+3\mu^2+\mu) = \frac{2^{3d}\mu^4(\mu^2+3\mu+1)}{n^3} .\end{aligned}$$

So the probability of such a bicycle is at most

$$\frac{2^{3d}\mu^4(\mu^2+3\mu+1)}{n^3}\left(\frac{\alpha_d\mu^2}{n}\right)^{L-2} = \frac{\mu^2+3\mu+1}{\mu^2}\left(\frac{2^d\mu^2}{n}\right)^{L+1}.$$

The three estimates prove the claim. ◀

Using the claim and summing from $L=1$ to R yields:

$$\sum_{L=1}^{R}\mathbb{E}Y_L \le \sum_{L=1}^{R}(2n)^L(2L)^2\frac{\mu^2+3\mu+1}{\mu^2}\left(\frac{2^d\mu^2}{n}\right)^{L+1} = O(n^{-1})$$

for any μ, R constant with respect to n. So whp there is no bicycle in the implication graph of length $\le R$ and thus no set of R literals that form an unsatisfiable formula.

For $k \ge 3$ consider an arrangement of R literals that yields an unsatisfiable k-SAT formula. The configuration of points would also induce an unsatisfiable 2-SAT formula since for each k-clause, each of the $\binom{k}{2}$ 2-clauses from the same set of literals would be present, and a satisfying assignment to the 2-SAT would also satisfy the k-SAT formula. But whp there is no set of R unsatisfiable 2-SAT literals, and so no set of R unsatisfiable k-SAT literals.

Condition 2: We want to show that there is no constant-sized set of positioned literals H, so that F_H is satisfiable but conditioning on the presence of H raises the probability of unsatisfiability of F_{G_μ} from α to $\alpha+\epsilon$ at the μ for which $\Pr[F_k(n,\mu)\notin SAT]=\alpha$. Assume $|H|\le R$. We will bound the conditional probability

$$\Pr[F_{G_\mu}\notin SAT | H\subseteq G_\mu] = \Pr[F_{G_\mu\cup H}\notin SAT]$$

where the equality follows from the properties of a Poisson point process. In other words, we will create a random formula by first placing the positioned literals in H in the cube, then adding each of the $2n$ literals independently on top according to a Poisson process of intensity μ, then forming the k-SAT formula from the entire set of points. Note that in the probability on the RHS H is a fixed point set, and G_μ a random point set that does not depend on H.

We now bound $\Pr[F_{G_\mu\cup H}\notin SAT]$. Let $\mathcal{X}_H$ be the set of variables of the literals in H. By assumption $|\mathcal{X}_H|\le kR$. First we show that whp the subformula of $F_{G_\mu\cup H}$ consisting of clauses entirely from $\mathcal{X}_H$ is satisfiable. By assumption, F_H is satisfiable so to create an unsatisfiable subformula on $\mathcal{X}_H$ we need the addition of G_μ to add at least one clause with variables entirely in $\mathcal{X}_H$. There are two different ways this could happen - either a clause is created entirely with randomly placed literals, or a clause is created with some literals from H and some random literals.

We bound the expected number of clauses in F_{G_μ} containing only variables from $\mathcal{X}_H$, call this $\mathbb{E}Y_{\mathcal{X}_H,\mu}$, by bounding the number of literals from $\mathcal{X}_H$ appearing within distance $n^{-1/d}$ of each other in G_μ:

$$\mathbb{E}Y_{\mathcal{X}_H,\mu} \le \binom{2kR}{2}\frac{2^d\mu^2}{n} = o(1).$$

Next, we bound the expected number of literals from $\mathcal{X}_H$ placed by G_μ within distance $n^{-1/d}$ of a literal in H. The total volume of the cube within distance $n^{-1/d}$ of H is bounded by $2^dk^2R^2/n$, and so the expected number of literals from $\mathcal{X}_H$ appearing at random in this region is bounded by $2^dk^2R^2(2kR\mu)/n = o(1)$.

The remainder of the proof follows the general plan of Section 5 of [10]. We separate the n variables into two sets $\mathcal{X}_H$ and $\mathcal{X}_H^c$, and we have shown that whp after the addition

of G_μ there is an assignment to $\mathcal{X}_H$ that satisfies the subformula of clauses entirely in $\mathcal{X}_H$, call this assignment x_H. We now show that with probability at least $1-\alpha-\epsilon/2$, we can extend this assignment on $\mathcal{X}_H^c$ to satisfy $F_{G_\mu \cup H}$. The remaining formula consists of two types of clauses: clauses which contain variables from $\mathcal{X}_H$ (overlapping clauses) and clauses that contain only variables from $\mathcal{X}_H^c$ (non-overlapping). With probability at least $1-\alpha$, the set of non-overlapping clauses in F_{G_p} is satisfiable, from the definition of μ. We will show that adding the overlapping clauses decreases this probability by at most $\epsilon/2$.

Step 1: The overlapping clauses created with the addition of G_μ are dominated (in terms of inducing unsatisfiability) by adding a constant number of independent random unit clauses.

We can assume that F_H is maximal in the sense that it admits exactly one satisfying assignment, x_H. Adding H to G_μ has two effects: it adds the constraint that $\mathcal{X}_H = x_H$ and it may create some new clauses involving positioned literals from H and G_μ. We have shown above that whp these new clauses all contain at least one variable from $\mathcal{X}_H^c$. Consider the following modification of F_{G_μ}: call the set of literals from $\mathcal{X}_H^c$ that fall within distance $n^{-1/d}$ of a literal from $\mathcal{X}_H$ (either in H or in G_μ) L. Note that the literals in L are uniformly random over all literals in $\mathcal{X}_H^c$. Remove the set L from G_μ to form the random point set G_μ^-. Create the formula $F^*_{G_\mu^-}$ by forming k-clauses according to the usual rules for G_μ^-, but add a unit clause (l) for every literal $l \in L$ that was removed from G_μ. Critically the k-clauses of $F^*_{G_\mu^-}$ are independent of the unit clauses of $F^*_{G_\mu^-}$ since they are formed from points from disjoint regions of the cube. Note that if there is a satisfying assignment to $F^*_{G_\mu^-}$, then the same assignment satisfies F_{G_μ}. The inequality goes in the correct way: we progress to a formula which has less probability of being satisfied.

The expected number of literals from G_μ that fall within distance $n^{-1/d}$ of a literal in $\mathcal{X}_H$ is bounded by $2^d/n \cdot (\mu+1)2kR(2n\mu) = 2^{d+2}kR\mu(\mu+1)$, so with probability $1-\epsilon/4$ the size of L is at most $2^{d+4}kR\mu(\mu+1)/\epsilon$.

Now consider the random formula F' which is formed by sampling a copy of F_{G_μ} and adding to it $2^{d+4}kR\mu(\mu+1)/\epsilon$ independent, uniformly random unit clauses from all $2n$ literals. With probability $1-o(1)$ this is the same as adding the same number of uniformly random unit clauses chosen from $\mathcal{X}_H^c$, and F_{G_μ} stochastically dominates the k-clauses of $F^*_{G_\mu^-}$ (formed from a Poisson process on a larger region), so $\Pr[F' \in SAT] \leq \Pr[F^*_{G_\mu^-} \in SAT] + \epsilon/4 \leq \Pr[F_{G_\mu \cup H} \in SAT] + \epsilon/4 + o(1)$.

Step 2: $\Pr[F' \in SAT] \geq \Pr[F_{G_\mu} \wedge C_1 \wedge \cdots \wedge C_{\sqrt{n}} \in SAT]$, where the C_i's are a collection of $\sqrt{n}$ independent, uniformly random k-clauses. This is Lemma 5.7 from [10].

Step 3: $\Pr[F_{G_\mu} \wedge C_1 \wedge \cdots \wedge C_{\sqrt{n}} \in SAT] \geq \Pr[F_{G_\mu \cup G_{\mu_s}} \in SAT]$, where G_{μ_s} is an independent sprinkling of random positioned literals with intensity $\mu_s = n^{-\delta}$ for each of the $2n$ literals. We will sprinkle literals independently, adding each literal as a Poisson process of intensity μ_s. Split the cube into n disjoint small cubes with side length $n^{-1/d}$. The probability that a single small cube has at least k sprinkled literals is $\sim (2\mu_s)^k/k! = 2^k n^{-k\delta}/k!$. The expected number of boxes with k literals is $\Theta(n^{1-k\delta})$ and whp there are at least $n^{1-2k\delta}$ such boxes. If we pick one k-clause at random from each box that has one, we will get a set of at least $n^{1-2k\delta}$ uniform and independent random k-clauses. Picking $\delta = 1/5k$ suffices.

Step 4: Increasing μ to $\mu' = \mu + \mu_s$ lowers the probability of satisfiability by at most $Cn^{-\delta} = Cn^{-1/5k}$, from the assumption of a coarse threshold (bounded derivative of the probability with respect to μ, $\mu \cdot d\Pr_\mu(\text{UNSAT})/d\mu \leq C$).

All together we have:

$$\begin{aligned}\Pr[F_{G_\mu} \in SAT | H \subseteq G_p] &\geq \Pr[F_{G_\mu \cup H} \in SAT] \\ &\geq \Pr[F^*_{G^-_\mu} \in SAT] + o(1) \\ &\geq \Pr[F' \in SAT] - \epsilon/4 + o(1) \\ &\geq \Pr[F_{G_\mu} \wedge C_1 \wedge \cdots \wedge C_{\sqrt{n}} \in SAT] - \epsilon/4 + o(1) \\ &\geq \Pr[F_{G_\mu \cup G_{\mu_s}} \in SAT] - \epsilon/4 + o(1) \\ &\geq \Pr[F_{G_\mu} \in SAT] - Cn^{-\delta} - \epsilon/4 + o(1)\end{aligned}$$

This contradicts condition 2 in Theorem 1, leading to the conclusion that the threshold must in fact be sharp.

6 Clause Density

The clause density in each k-SAT model is as follows:

▶ **Proposition 8.** *The number of clauses in $F_k(n,\gamma)$ is $\frac{2^k\gamma^{d(k-1)}k^d}{k!}n + o(n)$ whp. The number of clauses in $F_k(n,\mu)$ is $\frac{(2\mu)^k k^d}{k!}n + o(n)$ whp.*

Proof. Let X be the number of clauses in the random formula. To compute $\mathbb{E}X$, note that the probability that k given points, distributed uniformly at random in $[0,1]^d$ lie in an ℓ_∞-ball of diameter $\gamma n^{-1/d}$ is the probability that the smallest and largest of k independent uniform $[0,1]$ random variables differ by at most $\gamma n^{-1/d}$, raised to the dth power. This probability, p_k, can be computed by conditioning on the position of the smallest value:

$$\begin{aligned}p_k &= \int_0^1 k(1-t)^{k-1} \min\left\{1, \left(\frac{\gamma n^{-1/d}}{1-t}\right)^{k-1}\right\} dt \\ &= k(\gamma n^{-1/d})^{k-1} \int_0^{1-\gamma n^{-1/d}} dt + k\int_{1-\gamma n^{-1/d}}^1 (1-t)^{k-1} dt \\ &= \frac{k\gamma^{k-1}}{n^{(k-1)/d}}\left(1 - \frac{k-1}{k}\gamma n^{-1/d}\right) = \frac{k\gamma^{k-1}}{n^{(k-1)/d}}\left(1 + o(1)\right).\end{aligned}$$

So in the $F_k(n,\gamma)$ model,

$$\mathbb{E}X = \binom{2n}{k} p_k^d \sim \frac{2^k\gamma^{d(k-1)}k^d}{k!}n.$$

Standard estimates show that $\mathrm{var}(X) = O(n)$, and so Chebyshev's inequality gives

$$X = \frac{2^k\gamma^{d(k-1)}k^d}{k!}n + o(n)$$

whp.

The result for the $F_k(n,\mu)$ model follows from conditioning on the total number of literals that appear in the cube and applying the result for $F_k(n,\gamma)$. As this number is concentrated around its expectation, $2\mu n$, we have, whp,

$$X = \frac{(2\mu)^k k^d}{k!}n + o(n).$$

◀

7 A Coarse Threshold for $\tilde{F}(n, r)$

Here we show that the model $\tilde{F}(n, r)$ in which variables are placed in $[0, 1]^d$ and signs of clauses drawn uniformly at random has a coarse threshold.

▶ **Proposition 9.** *Let $r = \gamma n^{-\frac{U(k)}{d(U(k)-1)}}$, where $U(k)$ is the minimal number of variables u so that there exists an unsatisfiable k-SAT formula on u variables so that no two clauses share the same set of k variables. Then*

$$\lim_{n\to\infty} \Pr[\tilde{F}(n, r) \in SAT] = g(\gamma)$$

for a function $g(\gamma) \in (0, 1)$. Further, $\lim_{\gamma\to 0} g(\gamma) = 1$ and $\lim_{\gamma\to\infty} g(\gamma) = 0$.

Proof. Claim: $U(k) \leq (\ln 2)^{1/(k-1)}(2k)^{k/(k-1)}$. In particular, $U(k)$ is finite.

Proof: Let $u \geq (\ln 2)^{1/(k-1)}(2k)^{k/(k-1)}$. Now consider a random formula formed by taking a clause for each of the $\binom{u}{k}$ distinct sets of k variables from the set of variables $x_1, \ldots x_u$, and then assigning signs uniformly at random. The expected number of satisfying assignments is:

$$2^u(1 - 2^{-k})^{\binom{u}{k}} < 1$$

for our choice of u (using basic estimates). So there exists some unsatisfiable formula on u variables in which each clauses has a distinct set of variables.

Now we show that satisfiability undergoes a coarse threshold at $r = n^{-\frac{U(k)}{d(U(k)-1)}}$. The general idea of the proof is that for $r = \gamma n^{-\frac{U(k)}{d(U(k)-1)}}$, the probability that there is a set of $U(k)$ variables in a ball of diameter r is bounded away from 0 and 1. The probability that each such set forms an unsatisfiable formula is also bounded away from 0 and 1. We then show that for this choice of r, if there is no such set of variables, the formula is satisfiable whp.

For $r = \gamma n^{-\frac{U(k)}{d(U(k)-1)}}$ the expected number of sets of $U(k)$ variables that form an unsatisfiable formula tends to a constant as $n \to \infty$. To see this note that the expected number of sets of $U(k)$ variables that fall in a ball of diameter r is a constant, and that any such set of variables is unsatisfiable with probability at least $2^{-U(k)}$ from the definition of $U(k)$. To see that it is at most a constant, note that the expected number of connected components of $U(k)$ variables is constant. A modification of Theorem 3.4 of [19] shows that the number of such unsatisfiable sets of variables has a Poisson distribution asymptotically. The mean of this Poisson random variable tends to ∞ as $\gamma \to \infty$ and to 0 as $\gamma \to 0$. Finally, if there is no such set, then the formula is satisfiable whp, since whp the RGG for this r consists of connected components of size at most $U(k)$. For a component of size $< U(k)$, there must be a satisfying assignment, by the definition of $U(k)$. ◀

8 Bounds on the Satisfiability Threshold

For $k \geq 3$ we give bounds on the satisfiability threshold, showing in particular that the transition from almost certain satisfiability to almost certain unsatisfiability occurs when the number of clauses is linear in the number of variables:

▶ **Proposition 10.** *For all $k \geq 3$ there exist functions $\overline{\gamma}(k), \underline{\gamma}(k), \overline{\mu}(k), \underline{\mu}(k)$ so that for any $\epsilon > 0$,*

1. *For $\gamma < \underline{\gamma}(k) - \epsilon$, whp $F_k(n, \gamma) \in SAT$. For $\gamma > \overline{\gamma}(k) + \epsilon$, whp $F_k(n, \gamma) \notin SAT$.*
2. *For $\mu < \underline{\mu}(k) - \epsilon$, whp $F_k(n, \mu) \in SAT$. For $\mu > \overline{\mu}(k) + \epsilon$, whp $F_k(n, \mu) \notin SAT$.*

We can take $\underline{\gamma}(k) = 2^{-(1+1/d)}$, $\underline{\mu}(k) = 2^{-(d+1)/2}$, $\overline{\gamma}(k) = (k-1)^{1/d}$, *and* $\overline{\mu}(k) = k + \ln 2$. *In particular, all functions are independent of* n *and so the threshold for satisfiability occurs with a linear number of clauses.*

For $F_k(n, \gamma)$, the lower bound follows from the lower bound in Theorem 5. For the same set of points in the cube, form both the corresponding 2-SAT formula and the k-SAT formula. For each k-clause the 2-SAT formula will include each of the $\binom{k}{2}$ subclauses of length 2. If there is a satisfying assignment to the 2-SAT formula, the same assignment will satisfy the k-SAT formula.

For an upper bound, we will show that the probability that any assignment is satisfying is 0. Fix an assignment σ, and consider the set of n false literals under σ. Set $\gamma > (k-1)^{1/d} + \epsilon$. Tile $[0,1]^d$ by $(\lceil n^{1/d}/\gamma \rceil)^d$ boxes of side length $\gamma n^{1/d}$ (with boxes along the boundary possibly smaller). For large enough n (depending on ϵ), the number of boxes is strictly less than $n/(k-1)$. By the pigeonhole principle there must be a box with at least k points, and so with probability 1, an unsatisfied clause is formed. This is true for any set of n literals, and so with probability 1 there is no satisfying assignment.

For $F_k(n, \mu)$, the lower bound again follows from the $k = 2$ case and Theorem 5. For the upper bound, we bound the expected number of satisfying assignments. There are 2^n possible assignments, so it is enough to show that the probability a given assignment σ is satisfying is at most q^n for some $q < 1/2$ independent of n. Tile $[0,1]^d$ by n boxes of side length $n^{-1/d}$. The probability that there is no k-clause of negative literals under σ is bounded by the probability that none of these boxes contain k negative literals. The nodes in the different boxes are independent, so we need to show that for large enough μ, the probability there are fewer than k negative literals in a single cube of side length $n^{-1/d}$ is strictly less than $1/2$. The number of negative literals in a single such cube has distribution $\text{Poiss}(\mu)$. The median of a Poisson with mean λ is at least $\lambda - \ln 2$, so if we pick $\overline{\mu}(k) > k + \ln 2$, then $\Pr[\text{Poiss}(\mu) < k] < 1/2$ and via a first-moment argument whp $F_k(n, \mu)$ is unsatisfiable.

Acknowledgements. Will Perkins was supported in part by an NSF postdoctoral fellowship. The authors thank Alfredo Hubard for many interesting conversations on this topic and the anonymous referees for several helpful suggestions.

References

1 Dimitris Achlioptas and Ehud Friedgut. A sharp threshold for k-colorability. *Random Structures and Algorithms*, 14(1):63–70, 1999.

2 Noga Alon and Joel H Spencer. *The probabilistic method*, volume 57. Wiley-Interscience, 2004.

3 Deepak Bal. On sharp thresholds of monotone properties: Bourgain's proof revisited. *arXiv preprint arXiv:1302.1162*, 2013.

4 Paul Balister, Béla Bollobás, and Amites Sarkar. Percolation, connectivity, coverage and colouring of random geometric graphs. In *Handbook of Large-Scale Random Networks*, pages 117–142. Springer, 2008.

5 B. Bollobás, C. Borgs, J. T. Chayes, J. H. Kim, and D. B. Wilson. The scaling window of the 2-sat transition. *Random Structures & Algorithms*, 18(3):201–256, 2001.

6 Béla Bollobás and A. G. Thomason. Threshold functions. *Combinatorica*, 7(1):35–38, 1987.

7 Jean Bourgain and Gil Kalai. Threshold intervals under group symmetries. *Convex Geometric Analysis MSRI Publications Volume 34, 1998*, page 59, 1998.

8 V. Chvátal and B. Reed. Mick gets some (the odds are on his side). In *Foundations of Computer Science, 1992. Proceedings., 33rd Annual Symposium on*, pages 620–627. IEEE, 1992.

9 C. Cooper, A. Frieze, and G. B. Sorkin. A note on random 2-sat with prescribed literal degrees. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 316–320. Society for Industrial and Applied Mathematics, 2002.

10 E. Friedgut. Sharp thresholds of graph properties, and the k-sat problem. *Journal of the American Mathematical Society*, 12(4):1017–1054, 1999.

11 E. Friedgut, G. Kalai, et al. Every monotone graph property has a sharp threshold. *Proceedings of the American mathematical Society*, 124(10):2993–3002, 1996.

12 Ashish Goel, Sanatan Rai, and Bhaskar Krishnamachari. Sharp thresholds for monotone properties in random geometric graphs. In *STOC'04: Proceedings of the 36th Annual ACM Symposium on Theory of computing*, pages 580–586, New York, NY, USA, 2004. ACM Press.

13 P. Gupta and P. R. Kumar. Critical power for asymptotic connectivity. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 1, pages 1106–1110, 1998.

14 Zhenning Kong and Edmund M Yeh. Analytical lower bounds on the critical density in continuum percolation. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on*, pages 1–6. IEEE, 2007.

15 Gregory L. McColm. Threshold functions for random graphs on a line segment. *Combinatorics Probability and Computing*, 13(3):373–387, 2004.

16 Ronald Meester and Rahul Roy. *Continuum percolation*. Cambridge tracts in mathematics. Cambridge University Press, Cambridge, New York, 1996.

17 Andrea Montanari and Antoine Sinton. A simple one dimensional glassy kac model. *Journal of Statistical Mechanics: Theory and Experiment*, 2007(08):P08004, 2007.

18 Mathew D. Penrose. The Longest Edge of the Random Minimal Spanning Tree. *The Annals of Applied Probability*, 7(2):340–361, 1997.

19 Mathew D. Penrose. *Random Geometric Graphs*. Oxford University Press, 2003.

20 J. M. Schwarz and A. Alan Middleton. Percolation of unsatisfiability in finite dimensions. *Physical Review E*, 70(3):035103, 2004.

21 S. Torquato and M. D. Rintoul. Effect of the interface on the properties of composite media. *Physical Reviw Letters*, 75(22):4067–4070, 1996.

Average Case Polyhedral Complexity of the Maximum Stable Set Problem

Gábor Braun[1], Samuel Fiorini[2], and Sebastian Pokutta[1]

1 **ISyE, Georgia Institute of Technology, Atlanta, GA, USA**
{gabor.braun,sebastian.pokutta}@isye.gatech.edu

2 **Department of Mathematics, Université libre de Bruxelles CP 216, Bd. du Triomphe, 1050 Brussels, Belgium**
sfiorini@ulb.ac.be

Abstract

We study the minimum number of constraints needed to formulate random instances of the maximum stable set problem via LPs (more precisely, linear extended formulations), in two distinct models. In the uniform model, the constraints of the LP are not allowed to depend on the input graph, which should be encoded solely in the objective function. There we prove a $2^{\Omega(n/\log n)}$ lower bound with probability at least $1-2^{-2^n}$ for every LP that is exact for a randomly selected set of instances; each graph on at most n vertices being selected independently with probability $p \geqslant 2^{-\binom{n/4}{2}+n}$. In the non-uniform model, the constraints of the LP may depend on the input graph, but we allow weights on the vertices. The input graph is sampled according to the $G(n,p)$ model. There we obtain upper and lower bounds holding with high probability for various ranges of p. We obtain a super-polynomial lower bound all the way from $p = \Omega(\frac{\log^{6+\varepsilon} n}{n})$ to $p = O(\frac{1}{\log n})$. Our upper bound is close to this as there is only an essentially quadratic gap in the exponent, which also exists in the worst-case model. Finally, we state a conjecture that would close this gap, both in the average-case and worst-case models.

1998 ACM Subject Classification F.2.2 Computations on discrete structures, G.2.2 Graph algorithms, General terms: performance, theory

Keywords and phrases polyhedral approximation, extended formulation, stable sets

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.515

1 Introduction

In the last three years, extended formulations considerably gained interest in various areas, including discrete mathematics, combinatorial optimization, and theoretical computer science. The key idea underlying extended formulations is that with the right choice of variables, various combinatorial optimization problems can be *efficiently* expressed via linear programs (LPs). This asks for the intrinsic difficulty of expressing optimization problems through a single LP, in terms of the minimum number of necessary *constraints*. This leads to a complexity measure that we call loosely here 'polyhedral complexity' (precise definitions are given later in Section 2).

On the one hand, there is an ever expanding collection of examples of small size extended formulations. For instance, [16] has expressed the minimum spanning tree problem on a planar graph with only a linear number of (variables and) constraints, while in the natural edge variables the LP has an exponential number of constraints. There exist numerous other examples, see e. g., the surveys by [7] and [11].

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 515–530

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

On the other hand, a recent series of breakthroughs in lower bounds renewed interest for extended formulations [14, 9, 2, 5, 3, 6, 15]. These breakthroughs make it now conceivable to quantify the polyhedral complexity of any given combinatorial optimization problem *unconditionally*, that is, independently of conjectures such as P vs. NP, and without extra assumption on the structure of the LP.

Although a polynomial upper bound on the polyhedral complexity yields a polynomial upper bound on the true algorithmic complexity of the problem—provided that the LP can be efficiently constructed and also that the size of the coefficients is kept under control (see [14] for a discussion of this last issue) e. g., through interior point methods—it is becoming clear that the converse does not hold. Recently, [6] proved that every LP for MAXCUT with approximation factor at most $2-\varepsilon$ needs at least $n^{\Omega(\frac{\log n}{\log\log n})}$ constraints, while the approximation factor of the celebrated SDP-based polynomial time algorithm of [10] is close to 1.13. Even more recently, [15] solved another major open problem in the area by showing a $2^{\Omega(n)}$ lower bound on the size of any LP expressing the perfect matching problem and in [4] it was shown that the matching polytope does not admit any fully-polynomial size relaxation scheme (the polyhedral equivalent of an FPTAS).

In this paper, we consider the problem of determining the *average case* polyhedral complexity of the maximum stable set problem, in two different models: 'uniform' and 'non-uniform', see Section 1.2 below. Roughly, the uniform model asks for a single LP that works for a given set of input graphs. In the non-uniform model the LP can depend on the input graph G but should work for every choice of weights on the vertices of G (in particular, for all induced subgraphs of G).

We show that the polyhedral complexity of the maximum stable set problem remains high in each of these models, when the input graph is sampled according to natural distributions. Therefore, we conclude that *the hardness of the maximum stable set problem is not concentrated on a small mass of graphs but is spread out through all graphs.*

1.1 Related Work

Our work is most directly related to [9] and [2], where the framework for bounding the size of approximate linear programming formulations was laid out. This framework forms the basis of our uniform model. We will also employ a robustness theorem from [3] for dropping constraints and feasible solutions.

1.2 Contribution

We present the first strong and unconditional results on the average case size of LP formulations for the maximum stable set problem. In particular, we establish that the maximum stable set problem in two natural average case models and encodings does not admit a polynomial size linear programming formulation, even in the unlikely case that P = NP.

Uniform Model

In the *uniform model* the polytope P containing the feasible solutions to the stable set problem is *independent* of the instances. The instances will be solely encoded into the objective functions. This ensures that no complexity of the problem is leaked into an instance-specific formulation. A good example of a uniform model is the TSP polytope over K_n with which we can test for Hamiltonian cycles in any graph with at most n vertices by choosing an appropriate objective function. In the uniform model, we consider the random

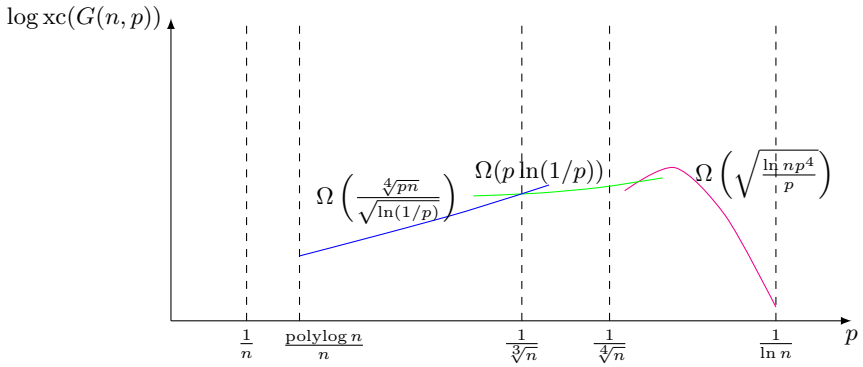


Figure 1 Comparing lower bounds on $\mathrm{xc}(G(n,p))$ for various regimes. For p close to $1/\sqrt[3]{n}$ the blue and green lines provide roughly the same bounds. For p significantly above $1/\sqrt[4]{n}$ the magenta line outperforms the green line.

instance set of graphs on at most n vertices, where each graph is contained in the instance set with probability $p \geqslant 2^{-\binom{n/4}{2}+n}$. Then we show that with probability at least $1-2^{-2^n}$, every LP formulation of this instance set in the natural encoding has at least $2^{\Omega(n/\log n)}$ constraints.

Non-uniform Model

In the *non-uniform model* we consider the stable set polytope for a *specific but random graph*. The polyhedral description may depend heavily on the chosen graph. We sample a graph G in the Erdős–Rényi $G(n,p)$ model, i. e., G has n vertices, and every pair of vertices is independently connected by an edge with probability p. We then analyze the stable set polytope $\mathrm{STAB}(G)$ of G. If p is small enough, so that the obtained graph is sufficiently sparse, it will contain an induced subgraph of sufficient size inducing a polyhedral reduction from the correlation polytope. Via this reduction we derive strong lower bounds on the size of any LP expressing $\mathrm{STAB}(G)$ that hold with high probability. In particular, we obtain superpolynomial lower bounds for p ranging between $\Omega(\frac{\log^{6+\varepsilon} n}{n})$ and $O(\frac{1}{\log n})$. For example for $p = n^{-\varepsilon}$ and $\varepsilon < 1/4$, any LP has at least $2^{\Omega(\sqrt{n^{\varepsilon}\log n})}$ constraints w.h.p., and for $p = \Omega(\frac{\log^{6+\varepsilon} n}{n})$, any LP has at least $n^{\log(3/2)\log^{\varepsilon/5} n}$ constraints w.h.p. Figure 1 illustrates our lower bounds.

1.3 Outline

In Section 2 we recall basics on extended formulations. We introduce the uniform model for the maximum stable set problem in Section 3.1. We then establish bounds on the average case complexity for the uniform model in Section 3. In Section 4 we consider the non-uniform model and derive lower bounds as well as upper bounds. We conclude with a conjecture in Section 5.

2 Preliminaries

We start by briefly recalling basics of extended formulations, stated in geometric terms. We refer the interested reader to [9] for more details. After that we state the main source of lower bounds in the non-uniform case.

Let $P \subseteq \mathbb{R}^d$ and $L \subseteq \mathbb{R}^e$ be two polyhedra. Then L is called an *extension* (or *lift*) of P if there exists an affine map $\pi\colon \mathbb{R}^d \to \mathbb{R}^e$, so that $\pi(L) = P$. Defining the *size* of polyhedron L as its number of facets, the *extension complexity* of polyhedron P is the minimum size of any of its extensions L, and is denoted by $\mathrm{xc}(P)$. Here we use the notions of extension and extended formulation interchangeably; the latter is simply an equivalent way to describe an extension.

The following monotonicity lemma from [9] provides a reduction mechanism to lower bound the extension complexity.

▶ **Lemma 1** (Monotonicity of extended formulations). *Let P be a polyhedron. Then the following hold:*

(i) *if F is a face of P, then $\mathrm{xc}(F) \leqslant \mathrm{xc}(P)$;*

(ii) *if L is an extension of P, then $\mathrm{xc}(P) \leqslant \mathrm{xc}(L)$.*

As usual, $\mathrm{corollary}(n) \coloneqq \mathrm{conv}\left(\{bb^\intercal \in \mathbb{R}^{n\times n} \,|\, b \in \{0,1\}^n\}\right)$ denotes the *correlation polytope* and $\mathrm{STAB}(G) \coloneqq \mathrm{conv}\left(\left\{\chi^S \in \mathbb{R}^{V(G)} \,\middle|\, S \text{ stable set of } G\right\}\right)$ is the *stable set polytope* of graph G. (Recall that the characteristic vector χ^S has $\chi^S_v = 1$ if $v \in S$ and $\chi^S_v = 0$ otherwise.) Let log denote the base-2 logarithm.

▶ **Theorem 2.** $\mathrm{xc}(\mathrm{corollary}(n)) \geqslant 2^{n\cdot\log(3/2)}$.

The factor $\log(3/2) \approx 0.585$ in the exponent is the current best one due to [12]; for various approximate case versions see [2, 5, 3]. The first exponential lower bound was established in [9].

The notion of extension directly generalizes to pairs of nested polyhedra. If $P \subseteq Q \subseteq \mathbb{R}^d$ are two polyhedra, an extension of the pair P, Q is a polyhedron $L \subseteq \mathbb{R}^e$ such that $P \subseteq \pi(L) \subseteq Q$ for some affine map $\pi\colon \mathbb{R}^d \to \mathbb{R}^e$. The extension complexity $\mathrm{xc}(P, Q)$ of pair P, Q is the minimum size of an extension of that pair.

3 Average Case Complexity in the Uniform Model

3.1 A Uniform Model for Maximum Stable Set

Faithful linear encodings were introduced in [2] to study the polyhedral hardness of approximation of various problems, via pairs of polyhedra. Here we recall only the polyhedral pair arising from the standard encoding of the maximum stable set problem.

We take $P = \mathrm{corollary}(n)$ as inner polyhedron, which is informally the convex hull of all possible vertex subsets, just like the 0/1-cube, but this encoding allows additionally to count edges by a linear function. In fact, a potential stable set I is encoded as the correlation matrix $bb^\intercal$ plays, where b is the characteristic vector of I. The outer polyhedron $Q = Q(\mathcal{G})$ depends on a collection $\mathcal{G}$ of graphs with vertex set included in $[n]$, that defines the instances we wish to solve. For each graph G with $V(G) \subseteq [n]$, we define an objective function $w^G \in \mathbb{R}^{n\times n}$ encoding the maximum stable set problem: i. e., assigning its size to every stable set of G, and carefully extended to other vertex subsets, so that the maximum is still the maximal size of stable sets of G. To this aim, we disregard vertices outside G and penalize edges inside the vertex set. This is formally achieved by letting $w^G_{ij} = 1$ if $i = j$ and $i \in V(G)$, but $w^G_{ij} = w^G_{ji} = -1$ if $ij \in E(G)$, and finally $w^G_{ij} = 0$ otherwise. We define a polyhedron

$$Q(\mathcal{G}) \coloneqq \left\{x \in \mathbb{R}^{n\times n}_+ \,\middle|\, \forall G \in \mathcal{G}\colon \langle w^G, x\rangle \leqslant \alpha(G)\right\},$$

of the constraints of maximum stable sets, where $\langle w^G, x\rangle = \sum_{i,j} w^G_{ij} x_{ij}$ denotes the Frobenius inner product of matrices w^G and x, and $\alpha(G)$ is the stability number of G. We let

$\mathrm{STAB}^u(\mathcal{G}, \rho)$ denote the pair of nested polyhedra $(P, (1+\rho)Q(\mathcal{G}))$ with $\rho \geqslant 0$ defining the dilation factor. If $\rho = 0$, we simply denote the pair by $\mathrm{STAB}^u(\mathcal{G})$.

For each polyhedron K containing P and contained in $(1+\rho)Q(\mathcal{G})$, we have $\max\{\langle w^G, x\rangle \mid x \in K\} \geqslant \alpha(G)$ for all graphs G, so that K is a relaxation of the maximum stable set problem. Moreover, $\max\{\langle w^G, x\rangle \mid x \in K\} \leqslant (1+\rho)\alpha(G)$ for all $G \in \mathcal{G}$. In other words, the relaxation K is not more than a $(1+\rho)$ factor off for all graphs in the collection $\mathcal{G}$.

The extension complexity of a polyhedral pair is equal to the nonnegative rank of any of its slack matrices up to a difference of 1, this is called the *factorization theorem*. For the pair $(P, (1+\rho)Q(\mathcal{G}))$ a slack matrix S has rows indexed by all the characteristic vectors $b \in \{0,1\}^n$ of the subsets of $[n]$, corresponding to the vertex $bb^\intercal$ of P, and columns indexed by $\mathcal{G}$. The entries are $S(G, b) = (1+\rho)\alpha(G) - \langle w^G, bb^\intercal\rangle$.

For example, when $\mathcal{G}$ is the set of all cliques, we may reindex the graphs by the characteristic vectors $a \in \{0,1\}^n$ of their vertex sets. We obtain a matrix M' as a slack matrix with rows and columns indexed by $a, b \in \{0,1\}^n$, and with entries $M'(a,b) = (1 - a^\intercal b)^2 + \rho$. Thus, in particular, restricting to the entries with $a^\intercal b \leq 1$, we obtain a partial matrix M

$$M(a,b) = \begin{cases} \rho & \text{if } a^\intercal b = 1 \\ 1+\rho & \text{if } a^\intercal b = 0. \end{cases}$$

For $\rho = 0$, the partial matrix M is known as the *unique disjointness (UDISJ)* (partial) matrix. For general $\rho \geqslant 0$, this is called the *ρ-shifted* UDISJ matrix. We shall need the following theorem from [3] to bound the nonnegative rank of certain submatrices of the (ρ-shifted) UDISJ matrix.

▶ **Theorem 3.** *For the ρ-shifted UDISJ matrix M, let M_k be the submatrix for sets of size k. Let S be any submatrix of M_k obtained by deleting at most an α-fraction of rows and at most a β-fraction of columns for some $0 \leqslant \alpha, \beta < 1$. Then for $0 < \varepsilon < 1$:*

$$\mathrm{rank}_+ S \geqslant 2^{(1/8(\rho+1) - (\alpha+\beta)\mathbb{H}[1/4])n - O(n^{1-\varepsilon})} \quad \textit{for} \quad k = n/4 + O(n^{1-\varepsilon}).$$

3.2 Average Case Complexity

We will now establish our main result for the uniform average case complexity model. We obtain that for any random collection of graphs where each graph is picked independently with probability p, the polyhedral complexity of solving the stable set problem over that particular collection of graphs is high, or more precisely, the extension complexity of the corresponding pair is high. This shows in particular that the instances of the stable set problem resulting in high extension complexity are not localized in a set of small density.

▶ **Main Theorem 4** (Super-polynomial xc of $\mathrm{STAB}^u(\mathcal{G})$ w.h.p.)**.** *Let $n \geqslant 40$ and $p \in [0,1]$ with $p \geqslant 2^{-\binom{n/4}{2}+n}$. Pick a random family $\mathcal{G}$ of graphs by adding each graph G with $V(G) \subseteq [n]$ to the family with probability p, independent of the other G. Then*

$$\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}^u(\mathcal{G})) \geqslant 2^{\Omega(n/\log n)}\right] \geqslant 1 - 2^{-2^n}.$$

A crucial point of the proof is a concentration result on $\alpha(G)$. It is well-known that almost all graphs G on n vertices have stability number $\alpha(G) \sim 2\log n$. However, the following rough estimate will be sufficient for our purpose, see e. g. [8, Proposition 11.3.4, page 304] for a proof.

▶ **Lemma 5.** *Let $n \geqslant 10$. The probability that a uniformly sampled random graph G with $V(G) = [n]$ has $\alpha(G) \geqslant 3 \log n$ is at most n^{-1}.*

We are ready to prove the main theorem of this section.

Proof of Main Theorem 4. The main idea of the proof is that, with large enough probability, we have $\max\left\{\langle w^K, x\rangle \,\middle|\, x \in Q(\mathcal{G})\right\} = O(\log n)$ for many cliques K with $V(K) \subseteq [n]$ and $\Theta(n)$ vertices. This implies that some slack matrix of the pair $\mathrm{STAB}^u(\mathcal{G})$ contains the $O(\log n)$-shifted UDISJ as a submatrix obtained by picking a large fraction of the rows (and all columns). We apply Theorem 3.

Consider a clique K with $V(K) \subseteq [n]$, and size $k := \lceil n/4 \rceil$. We say that a graph G is *good* for K if $V(G) = V(K)$ and $\alpha(G) \leqslant 3 \log n$. Clique K is said to be *good* if some graph $G \in \mathcal{G}$ is good for K. Otherwise, K is called *bad*.

We claim that, with high probability, the total fraction of bad cliques among all k-cliques K is at most $\alpha := 1/(24 \log n)$. By Lemma 5, the total number of graphs G with $V(G) = V(K)$ that are not good for a fixed k-clique K is at most $k^{-1} 2^{\binom{k}{2}}$. Thus

$$\begin{aligned}\mathbb{P}[K \text{ is bad}] &= \mathbb{P}[\mathcal{G} \text{ contains no good graph for } K] \\ &\leqslant (1-p)^{(1-k^{-1})2^{\binom{k}{2}}} \leqslant \mathrm{e}^{-p(1-k^{-1})2^{\binom{k}{2}}} \leqslant 2^{-\frac{9}{10} 2^n \log \mathrm{e}} \leqslant \alpha\, 2^{-2^n}.\end{aligned}$$

where the second inequality follows from $k \geqslant n/4 \geqslant 10$ and $p \geqslant 2^{-\binom{n/4}{2}+n}$. Let X denote the random variable with value the number of bad k-cliques K. By Markov's inequality,

$$\mathbb{P}\left[X \geqslant \alpha \binom{n}{k}\right] \leqslant 2^{-2^n}.$$

If clique K is good and G is a good graph for K, the inequality $\langle w^G, x\rangle \leqslant 3 \log n$ is valid for $Q(\mathcal{G})$. Thus the inequality $\langle w^K, x\rangle \leqslant 3 \log n$ is also valid for $Q(\mathcal{G})$, because $x \geqslant 0$ is valid for $Q(\mathcal{G})$, and $w^K \leqslant w^G$.

Suppose that the fraction of cliques K with $V(K) \subseteq [n]$ and size $k = \lceil n/4 \rceil$ that are bad is at most α. We have shown that this holds with probability at least $1 - 2^{-2^n}$. By what precedes, we can define a slack matrix for the pair $\mathrm{STAB}^u(\mathcal{G})$ that contains a $(3 \log n)$-shift of UDISJ with at most an α-fraction of the rows thrown away. From Theorem 3 (with $\beta = 0$) and from the factorization theorem, the extension complexity of the pair $\mathrm{STAB}^u(\mathcal{G})$ is at least $2^{(1/8(3\log n+1)-\alpha \mathbb{H}[1/4]) \cdot n - O(n^{1-\varepsilon})} = 2^{\Omega(n/\log n)}$. ◀

A close inspection of the proof of Main Theorem 4 shows that we can immediately apply the framework in [2] to obtain a lower bound on the average case *approximate* extension complexity. We obtain the following result. The proof is identical, except that we choose $\alpha := 1/24(1+\rho)\log n$, and the inequalities that yield the slack matrix are of the form $\langle w^K, x\rangle \leqslant (1+\rho)3 \log n$ for all good cliques K with $k = \lceil n/4 \rceil$ vertices, which are all valid for $(1+\rho)Q(\mathcal{G})$. A complete proof will be given in the full version of the paper.

▶ **Corollary 6** (Super-polynomial xc of $\mathrm{STAB}^u(\mathcal{G}, \rho)$ w.h.p.)**.** *As in Main Theorem 4, let $\mathcal{G}$ be a random family of graphs such that each graph G with $V(G) \subseteq [n]$ is contained in $\mathcal{G}$ with probability $p \geqslant 2^{-\binom{n/4}{2}+n}$ independent of the other graphs. Then the ρ-approximate pair $\mathrm{STAB}^u(\mathcal{G}, \rho)$ with $\rho \leqslant \frac{n^{1-\varepsilon}}{\log n}$ for some $0 < \varepsilon < 1/2$ has extension complexity $2^{\Omega(n^\varepsilon)}$, with probability at least $1 - 2^{-2^n}$.*

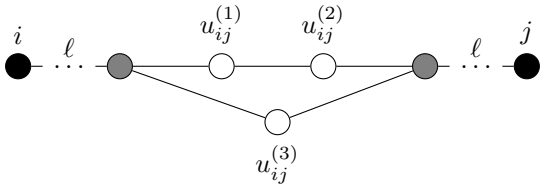


Figure 2 Edge gadget E_{ij} replacing edge ij of T in the gadget graph $T^{\pentagon}$. Black vertices represent vertices of the template graph T, white and grey vertices represent new vertices added to construct $T^{\pentagon}$. There are $\ell \geqslant 0$ many edges between the black and grey vertices.

Observe that the approximation factor in Corollary 6 can be larger than $3 \log n$. The reason why this is possible, contradicting initial intuition, is that the hardness arises from having many different graphs and hence many objective functions to consider simultaneously and the encoding is highly non-monotone. Roughly speaking, graphs with different vertex sets are independent of each other, even if one is an induced subgraph of the other.

4 Average Case Complexity in the Non-uniform Model

We now turn our attention to the non-uniform model, where we consider the stable set polytope over a *specific but random* graph G and analyze its extension complexity. Our strategy is to embed certain gadget graphs as induced subgraphs of G, using the probabilistic method. Here we consider the Erdős–Rényi graph model and sample G from $G(n,p)$.

We begin by defining the gadget graphs we use and seeing how an induced gadget forces up the extension complexity of $\mathrm{STAB}(G)$.

Fix a graph T. This graph serves as a template for defining the *gadget graph* of T, denoted as $T^{\pentagon}$: the graph obtained by replacing each edge ij of T with an *edge gadget* E_{ij}, which is a 5-cycle with two connecting paths (hairs) of length ℓ each, see Figure 2. In total, $T^{\pentagon}$ has $v := |V(T)| + (2\ell+3)\,|E(T)|$ vertices and $e := (2\ell+5)\,|E(T)|$ edges. We will always take ℓ even but allow $\ell = 0$, in which case i and j are part of the 5-cycle (on Figure 2, both i and j then coincide with the closest gray vertex).

The hairs are used to decrease the average degree of induced subgraphs of the gadget graph, which makes it easier to embed $T^{\pentagon}$ in $G(n,p)$ for lower values of p. This is formalized in the next lemma.

▶ **Lemma 7.** *For any graph T, the average degree of any induced subgraph of $T^{\pentagon}$ is at most $2 + 4/(2\ell+3)$ for $\ell \geqslant 1$. For $\ell = 0$ the average degree is at most 4.*

Proof. Let G be an induced subgraph of $T^{\pentagon}$. We shall prove the stronger claim that $|E(G)| \,/\, |V(G) \setminus V(T)|$ is upper bounded by 2 if $\ell = 0$ and by $1 + 2/(2\ell+3)$ if $\ell \geqslant 1$ (in other words, we ignore the original vertices of T at the estimation).

First we apply some modifications to G which do not decrease the factor $|E(G)| \,/\, |V(G) \setminus V(T)|$ if it was already at least 1. We add the original vertices of T to G (together with the edges connecting them to vertices already in G), and then we successively remove degree-1 vertices of G in the edge gadgets. Hence we may assume without loss of generality, that G has no degree-1 vertices of the edge gadgets, and it contains the original vertices of T. So for a fixed E_{ij}, the graph G contains either only the original vertices of T, or both the degree-3 (grey) vertices of the 5-cycle. In the latter case, from every path connecting these and i, j, the graph G contains either the whole path, or only the end points. We claim that if $\ell \geqslant 1$ then adding the missing paths will not decrease the factor $|E(G)| \,/\, |V(G) \setminus V(T)|$

below $1 + 2/(2\ell + 3)$ if it was greater than this value. Indeed, for every path, the ratio of added edges and vertices is at least $1 + 2/(2\ell + 3)$, namely, $1 + 1/(\ell - 1)$, 2 or 3/2. Therefore we may assume that every edge gadget E_{ij} is either completely contained in G or only the two vertices i and j of T are contained in G. Let k denote the number of E_{ij} completely contained in G, then $|E(G)| = k(2\ell + 5)$ and $|V(G) \setminus V(T)| = k(2\ell + 3)$, and their ratio is exactly $1 + 2/(2\ell + 3)$, finishing the proof in case $\ell \geqslant 1$.

If $\ell = 0$ then a similar argument applies, except that adding the shorter path (containing $u_{ij}^{(3)}$) and removing the longer path $(u_{ij}^{(1)}, u_{ij}^{(2)})$ will not decrease the factor $|E(G)| / |V(G) \setminus V(T)|$ below 2 if it were already larger. ◀

In the next lemma, we denote by corollary(T) the projection of the $|V(T)| \times |V(T)|$ correlation polytope corollary$(|V(T)|)$ on the variables x_{ii} for $i \in V(T)$ and x_{ij} for $ij \in E(T)$. We call this polytope the *correlation polytope of graph* T. In particular, corollary$(K_t) =$ corollary(t).

▶ **Lemma 8.** *If graph G contains $T^{\pentagon}$ (with arbitrary even hair length ℓ) as an induced subgraph, then*

$$\mathrm{xc}(\mathrm{STAB}(G)) \geqslant \mathrm{xc}(\mathrm{corollary}(T)).$$

In particular, for $T = K_t$ we get $\mathrm{xc}(\mathrm{STAB}(G)) \geqslant 2^{t \log(3/2)}$.

Proof. Let F be the face of STAB(G) whose vertices are the characteristic vectors of stable sets of $T^{\pentagon}$ containing the maximum number vertices in each edge gadget E_{ij}. Thus, F is defined by intersecting STAB(G) with the (face inducing) hyperplanes $\sum_{v \in V(E_{ij})} x_v = \ell + 2$ for all $ij \in E(T)$. Here x_v is the coordinate for vertex v in $T^{\pentagon}$. For simplicity, we denote by $x_{ij}^{(k)}$ the coordinate for the additional vertex $u_{ij}^{(k)}$ of the 5-cycle in E_{ij}, see Figure 2.

Then it can be easily verified that F is an extension of corollary(T) via the affine map $\pi \colon x \mapsto y = \pi(x)$ where

$$y_{ij} = \begin{cases} x_i & \text{if } i = j, \\ 1 - x_{ij}^{(1)} - x_{ij}^{(2)} & \text{if } i \neq j. \end{cases}$$

In this definition, the y_{ij} are the correlation variables, with $i, j \in V(T)$ and either $i = j$ or $ij \in E(T)$.

Now Lemma 1 gives

$$\mathrm{xc}(\mathrm{STAB}(G)) \geqslant \mathrm{xc}(F) \geqslant \mathrm{xc}(\mathrm{corollary}(T)).$$

For $T = K_t$, using Theorem 2, we have

$$\mathrm{xc}(\mathrm{STAB}(G)) \geqslant \mathrm{xc}(\mathrm{corollary}(t)) \geqslant 2^{t \log(3/2)}.$$

◀

4.1 Existence of Gadgets in Random Graphs

In this section, we estimate the probability that a random Erdős–Rényi graph $G = G(n, p)$ contains an induced copy of a graph H. Recall that in the $G(n, p)$ model, each of the $\binom{n}{2}$ pairs of vertices is connected by an edge with probability p, independently from the other edges. The next lemma is key for proving lower bounds on the extension complexity of STAB$(G(n, p))$ via embedding $H = T^{\pentagon}$ as an induced subgraph. The lemma is formulated in a general for future applications to many types of subgraphs H.

▶ **Lemma 9.** *Let H be a graph with v vertices and with all induced subgraphs having average degree at most d. Let $0 < p \leqslant 1/2$ and*

$$g = g(n,p,v) \coloneqq \frac{v^2 p^{-\frac{d}{2}}(1-p)^{-\frac{v}{2}}}{n-v}.$$

The probability of $G(n,p)$ not containing an induced copy of H satisfies

$$\mathbb{P}\left[H \overset{\text{ind}}{\not\subseteq} G(n,p)\right] \leqslant c_0 g^2 \approx 1.23 g^2,$$

where $c_0 \coloneqq \exp(2W(1/\sqrt{2}))/2$ and W is the Lambert W-function, the inverse of $x \to x \exp x$.

Proof. The proof is via the second-moment method.

Let S be any graph isomorphic to H with $V(S) \subseteq V(G)$. Let X_S be the indicator random variable of S being an induced subgraph of G. Obviously, the total number X of induced subgraphs of G isomorphic to H satisfies $X = \sum_S X_S$. We estimate the expectation and variance of X. Let e denote the number of edges of H, and let $\operatorname{Aut}(H)$ denote the automorphism group of H. The expectation is clearly

$$\mathbb{E}[X] = \sum_S \mathbb{E}[X_S] = \binom{n}{v} \frac{v!}{|\operatorname{Aut}(H)|} p^e (1-p)^{\binom{v}{2}-e}.$$

The variance needs more preparations. Let now S and T be two graphs isomorphic to H with $V(S), V(T) \subseteq V(G)$. Using that X_S and X_T are independent and thus $\operatorname{Cov}[X_S, X_T] = 0$ when $|V(S) \cap V(T)| \leqslant 1$ we get

$$\begin{aligned}\operatorname{Var}[X] &= \sum_{S,T} \operatorname{Cov}[X_S, X_T] \leqslant \sum_{|V(S)\cap V(T)|\geqslant 2} \mathbb{E}[X_S X_T] \\ &= \sum_{|V(S)\cap V(T)|\geqslant 2} \mathbb{E}[X_S]\,\mathbb{E}[X_T \mid X_S = 1] = \mathbb{E}[X] \sum_{T\colon |V(S)\cap V(T)|\geqslant 2} \mathbb{E}[X_T \mid X_S = 1].\end{aligned}$$

Note that in the last sum S is fixed, and by symmetry, the sum is independent of the actual value of S. That is why we could factor it out. We obtain via Chebyshev's inequality,

$$\mathbb{P}\left[H \overset{\text{ind}}{\not\subseteq} G(n,p)\right] = \mathbb{P}[X=0] \leqslant \frac{\operatorname{Var}[X]}{\mathbb{E}[X]^2} \leqslant \frac{\sum_{T\colon |V(S)\cap V(T)|\geqslant 2} \mathbb{E}[X_T \mid X_S = 1]}{\mathbb{E}[X]}.$$

We shall estimate $\mathbb{E}[X_T \mid X_S = 1]$, which is the probability that H is induced in G provided S is induced in G, as a function of $k \coloneqq |V(S) \cap V(T)|$. We assume that S and T coincide on $V(S) \cap V(T)$, and therefore have at most $dk/2$ edges in common, as their intersection is isomorphic to an induced subgraph of H, and therefore have average degree at most d by assumption. Hence as $p \leqslant 1/2$

$$\mathbb{E}[X_T \mid X_S = 1] = \mathbb{P}\left[T \overset{\text{ind}}{\subseteq} G \,\middle|\, S \overset{\text{ind}}{\subseteq} G\right] \leqslant p^{e-\frac{d}{2}k}(1-p)^{\binom{v}{2}-e-\binom{k}{2}+\frac{d}{2}k}.$$

This is clearly also true if S and T do not coincide on $V(S) \cap V(T)$, as then the probability

is 0. Now we can continue our estimation by summing up for all possible T with $k \geqslant 2$:

$$\frac{\sum_T \mathbb{E}\left[X_T \mid X_S = 1\right]}{\mathbb{E}\left[X\right]} \leqslant \frac{\sum_{k=2}^{v} \binom{v}{k}\binom{n-v}{v-k} \frac{v!}{|\operatorname{Aut} H|} p^{e-\frac{d}{2}k}(1-p)^{\binom{v}{2}-e-\binom{k}{2}+\frac{d}{2}k}}{\binom{n}{v}\frac{v!}{|\operatorname{Aut} H|}p^e(1-p)^{\binom{v}{2}-e}}$$

$$= \sum_{k=2}^{v} \frac{\binom{v}{k}\binom{n-v}{v-k}}{\binom{n}{v}} p^{-\frac{d}{2}k} \left(\underbrace{(1-p)^{\frac{d+1-k}{2}}}_{\leqslant (1-p)^{-\frac{v}{2}}}\right)^k \leqslant \sum_{k=2}^{v} \frac{v^k}{2(k-2)!}\left(\frac{v}{n-v}\right)^k \left(p^{-\frac{d}{2}}(1-p)^{-\frac{v}{2}}\right)^k$$

$$= \frac{1}{2}g^2 \sum_{k=2}^{v} \frac{1}{(k-2)!} g^{k-2} \leqslant \frac{1}{2}g^2 \exp(g),$$

as

$$\frac{\binom{v}{k}\binom{n-v}{v-k}}{\binom{n}{v}} \leqslant \frac{\binom{v}{k}\frac{(n-v)^{v-k}}{(v-k)!}}{\frac{(n-v)^v}{v!}} = \binom{v}{k}^2 \frac{k!}{(n-v)^k} \leqslant \frac{1}{k!}\left(\frac{v}{n-v}\right)^k.$$

The lemma follows: the probability of H not being an induced subgraph is at most $e^g g^2/2$. This upper bound is 1 exactly if $g = 2W(1/\sqrt{2})$. For $g \leqslant 2W(1/\sqrt{2})$, we obtain the upper bound in the lemma. For $g \geqslant 2W(1/\sqrt{2})$, the upper bound in the lemma is at least 1, so the statement is obvious. ◀

4.2 High Extension Complexity with High Probability

In order to obtain lower bounds on the extension complexity of the stable set polytope of $G = G(n,p)$, we use Lemma 9 together with Lemma 8, taking H to be $K_t^{\pentagon}$. We obtain the following result:

▶ **Main Theorem 10** (Super-polynomial xc of $\mathrm{STAB}(G(n,p))$ w.h.p.)**.** *With high probability, the stable set polytope of the random graph $G(n,p)$ has at least the following extension complexity, depending on the size of p:*

(i) *For $p = \omega(1/\sqrt[4]{n})$ and fixed $0 < c < 2/\sqrt{3} \approx 1.1547$, we have*

$$\mathbb{P}\left[\operatorname{xc}(\mathrm{STAB}(G(n,p))) \geqslant 2^{\sqrt{c\frac{\ln(np^4)}{p}}\log(3/2)}\right] = 1 - o(1). \tag{1}$$

(ii) *For $c > 0$ and $c/\sqrt[3]{n} \leqslant p = o(1)$ we have*

$$\mathbb{P}\left[\operatorname{xc}(\mathrm{STAB}(G(n,p))) \geqslant 2^{\frac{\log(3/2)}{\sqrt{p\ln(1/p)}}}\right] = 1 - O(1/c^6). \tag{2}$$

(iii) *Moreover, for any fixed $c > 0$ for all $1/n < p \leqslant c/\sqrt[3]{n}$ and $0 < \delta < 1$*

$$\mathbb{P}\left[\operatorname{xc}(\mathrm{STAB}(G(n,p))) \geqslant 2^{\delta\sqrt{\frac{\sqrt{pn}}{\ln(1/p)}}\log(3/2)}\right] \geqslant 1 - O(\delta^8). \tag{3}$$

As an illustration of Main theorem 10, we include concrete lower bounds in special cases of interest.

▶ **Corollary 11.** *For every fixed* $0 < \varepsilon < 1$*, we have*

$$\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G(n,n^{-\varepsilon}))) \geqslant 2^{\sqrt{(1-4\varepsilon)n^{\varepsilon}\ln n}\log(3/2)}\right] = 1-o(1) \qquad \text{for } \varepsilon < 1/4, \tag{4}$$

$$\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G(n,n^{-\varepsilon}))) \geqslant 2^{\frac{n^{\varepsilon/2}}{\sqrt{\varepsilon\ln n}}\log(3/2)}\right] = 1-o(1) \qquad \text{for } \varepsilon < 1/3, \tag{5}$$

$$\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G(n,n^{-\varepsilon}))) \geqslant 2^{\frac{n^{(1-\varepsilon)/4}}{\ln n}\log(3/2)}\right] = 1-o(1) \qquad \text{for } \varepsilon \geqslant 1/3. \tag{6}$$

Below the $p = n^{-\varepsilon}$ *range, we obtain*

$$\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G(n,(\ln^{6+\varepsilon} n)/n))) \geqslant 2^{\ln^{1+\varepsilon/5} n\cdot\log(3/2)}\right] = 1-o(1), \tag{7}$$

and (at the other end of the range) for fixed $\delta > 0$*,*

$$\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G(n,\delta\ln^{-1} n))) \geqslant n^{\delta^{-1/2}\log(3/2)}\right] = 1-o(1). \tag{8}$$

Proof. Equations (11) and (11) are special cases of (1). For Equation (11), we choose $p = n^{-\varepsilon}$ and $c = 1$. For Equation (1), we choose $p = \delta\ln^{-1} n$ and $c = 1.1$, a bit larger than 1, then the square root in (1) becomes

$$\sqrt{c\frac{\ln np^4}{p}} = \sqrt{c\frac{\ln n + 4\ln(\delta) + 4\ln\ln^{-1} n}{\delta}\ln n} = c\delta^{-1/2}(1+o(1))\ln n > \delta^{-1/2}\ln n,$$

proving the equation.

Equation (11) follows from Equation (2) via $p = n^{-\varepsilon}$. Equations (11) and (11) are special cases of Equation (3). Equation (11) is the case $p = n^{-1+\varepsilon}$ and $\delta = \sqrt{\varepsilon}\ln^{-1} n$. For Equation (11), we choose $p = (\ln^{6+\varepsilon} n)/n$ and $\delta = \ln^{-\varepsilon/20} n$, then the interesting part of the exponent is

$$\delta\sqrt{\frac{\sqrt{pn}}{\ln(1/p)}} = \ln^{-\varepsilon/20} n\sqrt{\frac{\sqrt{\ln^{6+\varepsilon} n}}{\ln n - \ln\ln^{6+\varepsilon} n}} > \ln^{-\varepsilon/20} n\sqrt{\frac{\sqrt{\ln^{6+\varepsilon} n}}{\ln n}} = \ln^{1+\varepsilon/5} n$$

proving the claim. ◀

Now we are going to prove the main theorem of Section 4.2.

Proof of Main Theorem 10. We apply Lemma 9 to the graph $H \coloneqq K_t^{\pentagon}$ together with Lemma 8 to obtain:

$$\begin{aligned}
\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G(n,p))) \geqslant 2^{t\log(3/2)}\right] &\geqslant \mathbb{P}\left[K_t^{\pentagon} \overset{\text{ind}}{\subseteq} G(n,p)\right] \\
&\geqslant 1 - c_0\,\frac{v^4 p^{-d}(1-p)^{-v}}{(n-v)^2} \\
&\geqslant 1 - c_0\,(1+o(1))\frac{v^4 p^{-d}\mathrm{e}^{pv}}{n^2} \qquad \text{if } v = o(n).
\end{aligned}$$

Here v is the number of vertices of H, and every induced subgraph of H should have average degree at most d. We shall estimate the last fraction $v^4 p^{-d}\mathrm{e}^{pv}/n^2$, using the d provided by Lemma 7. Below we will tacitly assume $t = \omega(1)$, which is w.l.o.g because $\mathrm{xc}(\mathrm{STAB}(G(n,p))) \geqslant n$ always.

Now we shall substitute various values for p, t, d, ℓ to obtain the equations of the theorem. We will verify $v = o(n)$ and $v^4 p^{-d} \mathrm{e}^{pv}/n^2 = o(1)$ to obtain an $1 - o(1)$ lower bound from the last inequality.

For establishing (1), we choose

$$\ell \coloneqq 0 \qquad\qquad t \coloneqq \left\lceil c\sqrt{\frac{\ln(np^4)}{p}} \right\rceil \qquad\qquad d \coloneqq 4.$$

Note that for $p \geqslant 1/\sqrt[4]{n}$,

$$v = t + 3\binom{t}{2} = \left(\frac{3}{2} + o(1)\right) t^2 = \left(\frac{3}{2} + o(1)\right) c^2 \frac{\ln(np^4)}{p} \leqslant \left(\frac{3}{2} + o(1)\right) c^2 \sqrt[4]{n} \ln n = o(n),$$

and hence

$$\begin{aligned} \frac{v^4 p^{-d} \mathrm{e}^{pv}}{n^2} &= \left(\frac{3}{2} + o(1)\right)^4 (pt^2)^4 \mathrm{e}^{(3/2+o(1))pt^2 - 2\ln(np^4)} \\ &\leqslant \left(\frac{3}{2} + o(1)\right)^4 c^8 \left(\ln(np^4)\right)^4 \exp\left\{\left[\left(\frac{3}{2} + o(1)\right) c^2 - 2\right] \ln(np^4)\right\} = o(1), \end{aligned}$$

as $np^4 = \omega(1)$ by assumption. This finishes the proof of (1).

We turn to (2) and (3). We will choose a positive ℓ to approximately minimize the fraction in terms of the other parameters. To ease computation, let

$$\gamma \coloneqq \frac{2\ell + 3}{2} > 1.$$

Then the parameters v and d look like

$$\begin{aligned} d &= 2 + \frac{4}{2\ell + 3} = 2 + \frac{2}{\gamma}, \\ v &= t + (2\ell + 3)\binom{t}{2} = \gamma t^2 + (1 - \gamma)t < \gamma t^2. \end{aligned}$$

Hence

$$\frac{v^4 p^{-d} \mathrm{e}^{pv}}{n^2} < \frac{\gamma^4 t^8 \mathrm{e}^{p\gamma t^2 + 2(\ln(1/p))/\gamma}}{p^2 n^2}.$$

The γ minimizing the expression is

$$\frac{\sqrt{4 + 2pt^2 \ln(1/p)} - 2}{pt^2} = \frac{2\ln(1/p)}{\sqrt{4 + 2pt^2 \ln(1/p)} + 2},$$

but we use an approximation as ℓ needs to be an even integer. Therefore we choose

$$\ell = 2\left\lceil \frac{\ln(1/p)}{\sqrt{4 + 2pt^2 \ln(1/p)} + 2} - \frac{3}{4} \right\rceil.$$

We will verify later that actually $\ell = \omega(1)$. Hence

$$\gamma = (1 + o(1)) \frac{2\ln(1/p)}{\sqrt{4 + 2pt^2 \ln(1/p)} + 2} = (1 + o(1)) \frac{\sqrt{4 + 2pt^2 \ln(1/p)} - 2}{pt^2},$$

and

$$\frac{v^4 p^{-d} \mathrm{e}^{pv}}{n^2} < (1+o(1)) \left(\frac{2pt^2 \ln(1/p)}{\sqrt{np^3}} \right)^4 \frac{\mathrm{e}^{(2+o(1))\sqrt{4+2pt^2\ln(1/p)}}}{\left(\sqrt{4+2pt^2\ln(1/p)}+2\right)^4}$$
$$= (1+o(1)) \frac{\mathrm{e}^{(2+o(1))\sqrt{4+2pt^2\ln(1/p)}} \left(\sqrt{4+2pt^2\ln(1/p)}-2\right)^4}{\left(np^3\right)^2}. \tag{9}$$

Now we shall substitute various values for p and t to obtain the equations of the theorem. We will need to verify $\ell = \omega(1)$ and $v = o(n)$ for every choice.

For Equation (3), i. e., in the case $1/n < p \leqslant c/\sqrt[3]{n}$, we neglect the exponential term in (4.2) for the choice of t:

$$t = \left\lceil \delta \sqrt{\frac{\sqrt{pn}}{\ln(1/p)}} \right\rceil.$$

Here $0 < \delta < 1$ is an additional parameter. Rearranging gives us

$$2pt^2 \ln(1/p) = (1+o(1))\delta^2 \sqrt{np^3} \leqslant (1+o(1))\delta^2 c^{3/2} \leqslant (1+o(1))c^{3/2},$$

so in particular,

$$\ell \geqslant 2 \left\lceil \frac{\ln(\sqrt[3]{n}/c)}{\sqrt{4+(1+o(1))c^{3/2}}+2} - \frac{3}{4} \right\rceil = \omega(1)$$
$$v < \gamma t^2 = O(1/p) = O(\sqrt[3]{n}) = o(n).$$

Finally,

$$\frac{v^4 p^{-d} \mathrm{e}^{pv}}{n^2} < (1+o(1)) \frac{\mathrm{e}^{(2+o(1))\sqrt{4+(1+o(1))c^{3/2}}} \left(\sqrt{4+(1+o(1))\delta^2\sqrt{np^3}}-2\right)^4}{\left(np^3\right)^2}$$
$$\leqslant (1+o(1)) \mathrm{e}^{(2+o(1))\sqrt{4+(1+o(1))c^{3/2}}} \left((1/4+o(1))\delta^2\right)^4 = O(\delta^8),$$

as claimed.

For Equation (2), i. e., when $c/\sqrt[3]{n} \leqslant p = o(1)$, we choose

$$t = \left\lceil \frac{1}{\sqrt{p\ln(1/p)}} \right\rceil.$$

This provides the estimate

$$2pt^2 \ln(1/p) = 2 + o(1),$$

hence $\ell = \Theta(\ln(1/p)) = \omega(1)$, and $v < \gamma t^2 = O(1/p) = O(\sqrt[3]{n}) = o(n)$. Finally,

$$\frac{v^4 p^{-d} \mathrm{e}^{pv}}{n^2} = (1+o(1)) \frac{\mathrm{e}^{(2+o(1))\sqrt{4+(2+o(1))}} \left(\sqrt{4+(2+o(1))}-2\right)^4}{\left(np^3\right)^2}$$
$$= O\left(\frac{1}{(np^3)^2}\right) = O(1/c^6),$$

as $np^3 \geqslant c^3$. ◀

Main Theorem 10 gives super-polynomial lower bounds all the way from $p = \Omega(\frac{\log^{6+\varepsilon} n}{n})$ to $p = O(\frac{1}{\log n})$. The key for being able to cover the whole regime is to have the gadgets depend on the parameter choice. Notice that for $p < 1/n$ a random graph almost surely will have all its components of size $O(\log n)$, making the stable set problem easy to solve, so that we essentially leave only a small polylog gap.

4.3 Upper Bound on Extension Complexity with High Probability

We now complement Main Theorem 10 with an upper bound, which is close to the lower bound, up to an essentially quadratic gap in the exponent.

▶ **Theorem 12** (Upper bound on the xc of $\mathrm{STAB}(G(n,p))$ w.h.p.). *For $0 < p \leqslant 1/2$,*

$$\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G)) \geqslant 2^{\Omega\left(\frac{\ln^2 n}{p}\right)}\right] \leqslant n^{-\Omega\left(\frac{\ln n}{p}\right)}.$$

In particular, for $p = n^{-\varepsilon}$, we obtain $\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G)) \geqslant 2^{\Omega\left(n^{\varepsilon}\ln^2 n\right)}\right] = o(1)$ and similarly for $p = \delta \ln^{-1} n$, we get $\mathbb{P}\left[\mathrm{xc}(\mathrm{STAB}(G)) \geqslant n^{\Omega\left(\frac{\ln^3 n}{\delta}\right)}\right] = o(1)$.

The upper bound stated in Theorem 12 essentially relies on the following basic result.

▶ **Lemma 13.** *Every polytope P has an extension complexity at most the number of its vertices.*

Proof. Let V be the set of vertices of P, and let Q be a simplex with $|V|$ vertices. The simplex Q is an extension of P via mapping the vertices of Q one-to-one to V in an arbitrary fashion, and extending to an affine mapping on Q. This extension has size $|V|$. ◀

We are ready to prove our theorem.

Proof of Theorem 12. By standard arguments (see, e. g., [8, Chapter 11, page 300]), for $G = G(n,p)$ we have

$$\mathbb{P}\left[\alpha(G) \geqslant r\right] \leqslant \left(n\,\mathrm{e}^{-p(r-1)/2}\right)^r$$

and thus for $r = 4\frac{\ln n}{p}$ we get

$$\mathbb{P}\left[\alpha(G) \geqslant 4\frac{\ln n}{p}\right] \leqslant \left(\frac{n}{\sqrt{e}}\right)^{-4\frac{\ln n}{p}}.$$

Therefore, with very high probability, we have $\alpha(G) \leqslant 4\frac{\ln n}{p}$. Using the inequality $\sum_{i=0}^{k}\binom{n}{i} \leqslant (n+1)^k$, we get

$$\#(\text{stable sets in } G) \leqslant (n+1)^{\alpha(G)} = 2^{\log(n+1)\alpha(G)} = 2^{\left(\frac{1}{\ln 2}+o(1)\right)\ln(n)\alpha(G)}.$$

The result then follows directly from Lemma 13. ◀

5 Concluding Remarks

We conclude with the following conjecture whose validity, we believe, is necessary to strengthen the result, close the remaining gap, as well as establishing truly exponential lower bounds on the extension complexity of further combinatorial problems.

▶ **Conjecture 14** (Sparse Graph Conjecture). *There exists an infinite family $(T_k)_{k\in\mathbb{N}}$ of template graphs such that, denoting by t_k the number of vertices of T_k: (i)* $\mathrm{xc}(\mathrm{corollary}(T_k)) = 2^{\Omega(t_k)}$*; (ii) T_k has bounded average degree; (iii) $t_k \leqslant t_{k+1}$ but at the same time $t_{k+1} = O(t_k)$.*

The existence of such a family would have various consequences.

Exact Case

Assuming the Sparse Graph Conjecture we would obtain that the extension complexity of polytopes for important combinatorial problems considered in [9, 1, 13] including (among others) the stable set polytope, knapsack polytope, and the 3SAT polytope would have truly exponential extension complexity, that is $2^{\Omega(n)}$ extension complexity, where n is the *dimension* of the polytope.

The recent groundbreaking result of [15] gives $2^{\Omega(n)}$ bounds for the extension complexity of the matching polytope and TSP polytope. These bounds are also tight up to constants, but this time the upper bound does not come from the number of vertices but rather from the number of facets and dynamic programming algorithms, respectively. Notice that the dimension of both polytopes is $d = \Theta(n^2)$, thus the bounds are in fact $2^{\Omega(\sqrt{d})}$.

Average Case

As observed above, there is a quadratic gap in the best current lower and upper bounds on the worst-case extension complexity of the stable set polytope: $2^{\Omega(\sqrt{n})}$ versus 2^n respectively. This is reflected in the results we obtain here. Assuming the Sparse Graph Conjecture we could reduce the gap between upper and lower bounds to a logarithmic factor. Moreover, our results could be strengthened to establish super-polynomial lower bounds on the average-case extension complexity up to constant probability p.

Acknowledgements. Research reported in this paper was partially supported by NSF grant CMMI-1300144.

References

1 D. Avis and H. R. Tiwary. On the extension complexity of combinatorial polytopes. *ArXiv e-prints*, February 2013.

2 G. Braun, S. Fiorini, S. Pokutta, and D. Steurer. Approximation Limits of Linear Programs (Beyond Hierarchies). In *53rd IEEE Symp. on Foundations of Computer Science (FOCS 2012)*, pages 480–489, 2012.

3 G. Braun and S. Pokutta. Common information and unique disjointness. In *IEEE 54th Annual Symp. on Foundations of Computer Science (FOCS 2013)*, pages 688–697, 2013. `http://eccc.hpi-web.de/report/2013/056/`.

4 G. Braun and S. Pokutta. The matching polytope does not admit fully-polynomial size relaxation schemes. *preprint available at* `http://arxiv.org/abs/1403.6710`, 2014.

5 M. Braverman and A. Moitra. An information complexity approach to extended formulations. In *Proceedings of the 45th annual ACM symposium on Theory of computing*, pages 161–170, 2013.

6 S. O. Chan, J. R. Lee, P. Raghavendra, and D. Steurer. Approximate constraint satisfaction requires large LP relaxations. In *IEEE 54th Annual Symp. on Foundations of Computer Science (FOCS 2013)*, pages 350–359, 2013.

7 M. Conforti, G. Cornuéjols, and G. Zambelli. Extended formulations in combinatorial optimization. *4OR*, 8:1–48, 2010.

8 R. Diestel. *Graph Theory*. Springer-Verlag Heidelberg, New York, 2005.

9 S. Fiorini, S. Massar, S. Pokutta, H. R. Tiwary, and R. de Wolf. Linear vs. Semidefinite Extended Formulations: Exponential Separation and Strong Lower Bounds. *Proc. STOC 2012*, 2012.

10 M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. Assoc. Comput. Mach.*, 42:1115–1145, 1995.

11 V. Kaibel. Extended formulations in combinatorial optimization. *Optima*, 85:2–7, 2011.

12 V. Kaibel and S. Weltge. A Short Proof that the Extension Complexity of the Correlation Polytope Grows Exponentially. *ArXiv e-prints*, July 2013.

13 S. Pokutta and M. Van Vyve. A note on the extension complexity of the knapsack polytope. *Operations Research Letters*, 41:347–350, 2013.

14 T. Rothvoß. Some 0/1 polytopes need exponential size extended formulations, 2011. arXiv:1105.0036.

15 Thomas Rothvoß. The matching polytope has exponential extension complexity. *ArXiv e-prints*, 2013.

16 J. C. Williams. A linear-size zero-one programming model for the minimum spanning tree problem in planar graphs. *Networks*, 39:53–60, 2002.

An Optimal Algorithm for Large Frequency Moments Using $O(n^{1-2/k})$ Bits*†

Vladimir Braverman[1], Jonathan Katzman[2], Charles Seidell[3], and Gregory Vorsanger[4]

1 **Johns Hopkins University, Department of Computer Science**
vova@cs.jhu.edu
2 **Johns Hopkins University**
jkatzma2@jhu.edu
3 **Johns Hopkins University**
cseidel5@jhu.edu
4 **Johns Hopkins University, Department of Computer Science**
gregvorsanger@jhu.edu

Abstract

In this paper, we provide the first optimal algorithm for the remaining open question from the seminal paper of Alon, Matias, and Szegedy: approximating large frequency moments. Given a stream $D = \{p_1, p_2, \ldots, p_m\}$ of numbers from $\{1, \ldots, n\}$, a frequency of i is defined as $f_i = |\{j : p_j = i\}|$. The k-th *frequency moment* of D is defined as $F_k = \sum_{i=1}^{n} f_i^k$.

We give an upper bound on the space required to find a k-th frequency moment of $O(n^{1-2/k})$ bits that matches, up to a constant factor, the lower bound of [48] for constant ϵ and constant k. Our algorithm makes a single pass over the stream and works for any constant[1] $k > 3$. It is based upon two major technical accomplishments: first, we provide an optimal algorithm for finding the heavy elements in a stream; and second, we provide a technique using Martingale Sketches which gives an optimal reduction of the large frequency moment problem to the all heavy elements problem. Additionally, this reduction works for any function g of the form $\sum_{i=1}^{n} g(f_i)$ that requires sub-linear polynomial space, and it works in the more general turnstile model. As a result, we also provide a polylogarithmic improvement for frequency moments, frequency based functions, spatial data streams, and measuring independence of data sets.

1998 ACM Subject Classification F.2 Analysis of Algorithms And Problem Complexity

Keywords and phrases Streaming Algorithms, Randomized Algorithms, Frequency Moments, Heavy Hitters

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.531

1 Introduction

The analysis of massive data sets has become an exciting topic of theoretical algorithms research. As these datasets grow increasingly large, the need to develop new algorithms which can run using sublinear memory has become paramount. It is often convenient to view such datasets as *data streams*. In this paper we consider the following streaming model:

* This work was supported in part by DARPA grant N660001-1-2-4014. Its contents are solely the responsibility of the authors and do not represent the official view of DARPA or the Department of Defense.
† This work was supported in part by Pistritto Fellowship.
[1] We stress that our bound only holds for $k = O(1)$. In fact, the dependence on k is at least exponential. See Section 1.1.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 531–544

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

▶ **Definition 1.** Let m and n be positive integers. A **stream** $D = D(n, m)$ is a sequence of integers $p_1, \ldots, p_m$, where $p_i \in \{1, \ldots, n\}$. A **frequency vector** is a vector of dimensionality n with non-negative entries $f_i, i \in [n]$ defined as:

$$f_i = |\{j : 1 \leq j \leq m, p_j = i\}|$$

A k-th **frequency moment** of a stream D is defined by $F_k(D) = \sum_{i \in [n]} f_i^k$. Also, $F_\infty = \max_{i \in [n]} f_i$ and $F_0 = |\{i : f_i > 0\}|$.

In their celebrated paper, Alon, Matias, and Szegedy [1, 1] introduced the following problem:

▶ **Problem 2.** *What is the space complexity of computing a $(1 \pm \epsilon)$-approximation of F_k in one pass over D?*

In this paper we consider the case where $k > 2$. Many algorithms have been designed to solve this particular problem, and we now provide a brief overview of the upper and lower bounds provided. To begin, [1] gave a lower bound of $\Omega(n^{1-5/k})$ (for $k \geq 6$) and an upper bound of $O(\frac{1}{\epsilon^2} n^{1-1/k} \log(nm))$. Bar-Yossef, Jayram, Kumar, and Sivakumar [4] improved the lower bound and showed a bound of $\Omega(n^{1-(2+\lambda)/k})$ for their one pass algorithm where λ is a small constant. They also showed a lower bound of $\Omega(n^{1-3/k})$ for a constant number of passes. Chakrabarti, Khot, and Sun [19] showed a lower bound of $\Omega(n^{1-2/k})$ for one pass and $\Omega(n^{1-2/k}/(\log n))$ for a constant number of passes. Gronemeier [31] and Jayram [36] extended the bound of [19] from one pass to multiple passes. Woodruff and Zhang [48] gave a lower bound of $\Omega(n^{1-2/k}/(\epsilon^{(4/p)}t))$ for a t-pass algorithm. Ganguly [28] improved the result of [48] for small values of ϵ and for $t = 1$. Price and Woodruff [44] gave a lower bound on the number of linear measurements and Andoni, Nguyen, Polyanskiy, and Wu [3] showed that $\Omega(n^{1-2/k} \log n)$ linear measurements are necessary.

In terms of upper bounds, Ganguly [26] and Coppersmith and Kumar [20] simultaneously gave algorithms with space complexity[2] $\tilde{O}(n^{1-1/(k-1)})$. In their breakthrough paper, Indyk and Woodruff [33] gave the first upper bound that is optimal up to a polylogarithmic factor. Their bound was improved by a polylogarithmic factor by Bhuvanagiri, Ganguly, Kesh, and Saha [7]. Monemizadeh and Woodruff [41] gave a bound of $O(\epsilon^{-2}k^2n^{1-2/k}\log^5(n))$ for a $\log(n)$-pass algorithm. For constant ϵ, Braverman and Ostrovsky [13] gave a bound of $O(n^{1-2/k}\log^2(n)\log^{(c)}(n))$ where $\log^{(c)}(n)$ is the iterated logarithm function. Andoni, Krauthgamer, and Onak [2] gave a bound of $O(k^2\epsilon^{-2-6/p}n^{1-2/k}\log^2(n))$. Ganguly [27] gave a bound of
$O(k^2\epsilon^{-2}n^{1-2/k}E(k,n)\log(n)\log(nmM)/\min(\log(n), \epsilon^{4/k-2}))$ where $E(k,n) = (1-2/k)^{-1}(1-n^{-4(1-2/k)})$. Braverman and Ostrovsky [16, 15] gave a bound of $O(n^{1-2/k}\log(n)\log^{(c)}(n))$.

1.1 Main Result

For constant ϵ and k we provide a streaming algorithm with space complexity $O(n^{1-2/k})$. Thus, our upper bound matches the lower bound of Woodruff and Zhang [48] up to a constant factor. Our algorithm makes a single pass over the stream and works for constant $k > 3$.

The main technical contribution is a new algorithm that finds heavy elements in a stream of numbers. Then, combining this result with the Martingale Sketches technique we create an algorithm to approximate F_k. In particular, we show:

[2] The standard notation $\tilde{O}$ hides factors that are polylogarithmic in terms of n, m and polynomial in terms of the error parameter ϵ.

▶ **Theorem 3.** *Let ϵ be a constant and $k \geq 7$. There exists an algorithm that outputs a $(1 \pm \epsilon)$-approximation of F_k, makes three passes over the stream, uses $O(n^{1-2/k})$ memory bits, and errs with probability at most $1/3$.*

We now present the necessary definitions and theorems.

▶ **Definition 4.** Let D be a stream and ρ be a parameter. The index $i \in [n]$ is a *ρ-heavy element* if $f_i^k \geq \rho F_k$.

▶ **Definition 5.** A randomized streaming algorithm $\mathcal{A}$ is an *Algorithm for Heavy Elements (AHE)* with parameters ρ and δ if the following is true: $\mathcal{A}$ makes three passes over stream D and outputs a sequence of indices and their frequencies such that if element i is a ρ-heavy element for F_k then i will be one of the indices returned[3]. $\mathcal{A}$ errs with probability at most δ.

▶ **Theorem 6.** *Let $k \geq 7$. There exists an absolute constant $C \leq 10$ and an AHE algorithm with parameters ρ and δ that uses*

$$O(\frac{1}{\rho^C}(F_0(D))^{1-2/k} \log \frac{1}{\delta}) \tag{1}$$

bits.

▶ **Theorem 7.** *Given Theorem 6, for any ϵ there exists an algorithm that uses*

$$O(\frac{1}{\epsilon^{2C}}(F_0(D))^{1-2/k}) \tag{2}$$

memory bits, makes three passes over D, and outputs a $(1 \pm \epsilon)$-approximation of F_k with probability at least $2/3$. Here C is the constant from Theorem 6.

From here, we see that the main theorem, Theorem 3, follows directly from Theorem 7.

After establishing the matching bound with three passes, we improve our algorithm further:

▶ **Theorem 8.** *Let ϵ be a constant and $k > 3$. Assuming that m and n are polynomially far, there exists an algorithm that outputs a $(1 \pm \epsilon)$-approximation of F_k, makes one pass over the stream, uses $O(n^{1-2/k})$ memory bits, and errs with probability at most $1/3$.*

We stress that our bound $O(n^{1-2/k})$ only holds for $k = O(1)$. In fact, the dependence on k is at least exponential. This is because we (initially) assume that for the heavy element f_1 it is true that $f_1^k \geq C \sum_{i>1} f_i^k$ for some large constant $C \geq 2^\Psi$ where $\Psi = \Omega(k)$. Later, we use a standard hashing technique to find heavy elements $f_1^k > \rho F_k$ for smaller values of ρ.

Additional Results

The previous theorems demonstrate the optimal reduction from the problem of computing frequency moments for constant $k > 2$ to the problem of finding heavy elements with constant error. The Martingale Sketches technique is an improvement over the previous method of recursive sketches [16]. Thus, our method is applicable in a general setting of approximating L_1-norms of vectors which have entries obtained by applying entry-wise functions on the frequency vector. As a result, we answer the main open question from [16] and improve several applications in [16].

[3] Indices of non-heavy elements can be reported as well.

Additionally, this reduction works for any function g of the form $\sum_{i=1}^{n} g(f_i)$ that requires sub-linear polynomial space, and it works in the more general turnstile model. As a result, we also provide a polylogarithmic improvement for frequency moments, frequency based functions, spatial data streams, and measuring independence of data sets. We will provide a detailed list of these results in the full version of the paper.

1.2 Related Work

Approximating F_k has become one of the most inspiring problems in streaming algorithms. To begin, we provide an incomplete list of papers on frequency moments [32, 25, 1, 14, 8, 11, 4, 12, 19, 5, 33, 20, 22, 24, 26, 29, 39, 13, 37, 38, 43, 46, 6, 18, 34, 27, 28, 48, 35] and references therein. These and other papers have produced many beautiful results, important applications, and new methods. Below we will mention a few of the results that provide relevant bounds. We refer a reader to [42, 47] and references therein for further details.

In [1], the authors observed that it is possible to approximate F_2 in optimal polylogarithmic space. Kane, Nelson and Woodruff [38] gave a space-optimal solution for F_0. Kane, Nelson, and Woodruff [37] gave optimal-space results for $F_k, 0 < k < 2$. In addition to the original model of [1], a variety of different models of streams have been introduced. These models include the turnstile model (that allows insertion and deletion) [32], the sliding window model [10, 23, 17], and the distributed model [30, 48, 21]. In the turnstile model, where the updates can be integers in the range $[-M, M]$, the latest bound by Ganguly [27] is

$$O(k^2\epsilon^{-2}n^{1-2/k}E(k,n)\log(n)\log(nmM)/\min(\log(n),\epsilon^{4/k-2}))$$

where $E(k,n) = (1-2/k)^{-1}(1-n^{-4(1-2/k)})$. Recently, Li and Woodruff provided a matching lower bound for $\epsilon < 1/(\log n)^{O(1)}$ [40]. The bound from [27] is roughly $O(n^{1-2/k}\log^2(n))$ for constant ϵ, k and it matches the earlier result of Andoni, Krauthgamer, and Onak [2]. For the turnstile model, the problem has been solved optimally for $\epsilon < 1/(\log n)^{O(1)}$ [27, 40]. These results combined with our result demonstrate that the turnstile model is fundamentally different from the model of Alon, Matias, and Szegedy.

2 Intuition

In this abstract we will provide a detailed but high-level explanation of the algorithms and techniques that we employ in the proof of our main theorem. All of this description, as well as the analysis required for a rigorous proof of this theorem, can be found in more detail in the full version [9].

2.1 High Level Description of the Algorithm

We present a composite algorithm to estimate frequency moments. At the absolute lowest degree of detail, we perform three steps. First, we determine the length of the stream. Second, we use a new algorithm to efficiently find heavy elements. Finally, we use a new technique to estimate the value of frequency moments from the weight of the found heavy elements. We now describe the intuition of each of these parts in detail.

2.2 The Heavy Hitter Algorithm

The key step in our algorithm for frequency moment computation is a new technique to compute the heavy hitters of a stream. In order to determine which elements are ρ-heavy

in stream D, we present an algorithm that is implemented as a sequence of sub-algorithms, and in general we will refer to each of these sub-algorithms as a "game". In [15] it is shown that $O(n^{1-2/k})$ samples are sufficient to solve the problem. However, each sample requires $\log n$ bits for counting the frequency (counters) and for identifying the elements (IDs). The resulting bound is $O(n^{1-2/k} \log n)$ bits. The goal of our algorithm, therefore, is to reduce the space required for counters and IDs from $\log n$ to an amortized $O(1)$ bits, achieving the optimal bound.

First we will describe the workings of a single instance of the game, and then we will describe the sequence of games that composes our heavy hitter algorithm. Each game in the sequence will be run in parallel, and the cost of the sequence of games will form a geometric series, which when evaluated will yield a total cost of $O(n^{1-2/k})$ bits. The crucial observation is that a heavy element in the stream will be returned by at least one of these games with constant probability, and will be sufficiently frequent to stand out from the other returned values as the true heavy hitter.

2.3 The Initial Algorithm

We will begin our solution by designing an algorithm for finding heavy elements in a stream which conforms to several assumptions. The key step of the algorithm is a subroutine that we call a *game*, which is described next. The algorithm will execute (in parallel) a sequence of several games with different parameters α and β.

The winner of each (α, β)-game will compete against the others, and the overall winner will be the output of the algorithm.

2.4 The Game

To find a heavy element of a stream and prove Theorem 6, we play a game using the stream as input. First we split the stream into equally sized rows as we read it in, and assemble them into a matrix M.

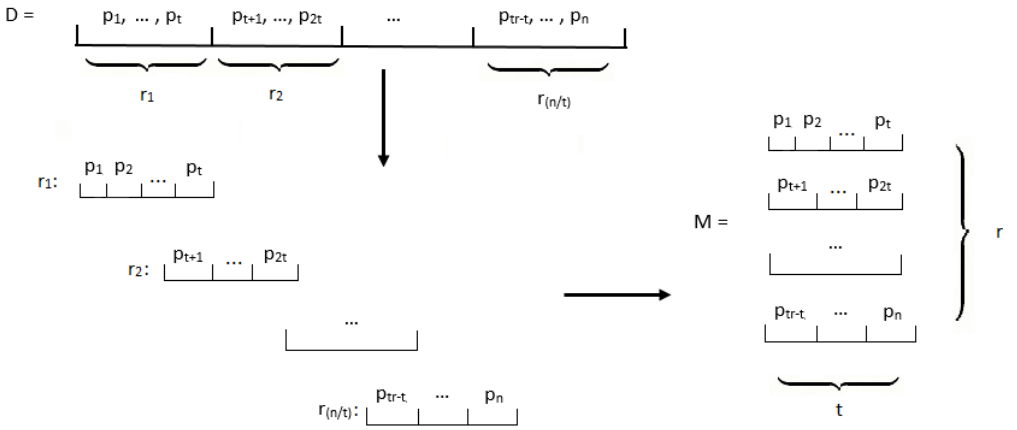


Figure 1 Transforming the stream into a matrix.

A single game is described colloquially as follows: for each row, we create a "team" that is composed of a group of $w = O(n^{1-2/k})$ players each competing to be the winner of that game. To create these teams, we sample elements from the current row to act as the players on each team, and give each player an ID number equal to one of the sampled elements. For

each player on a team, maintain a counter to track how often their ID number appears as we move through the stream. If the player's counter does not grow fast enough, that player is removed from the game.

The γ-th round is played by each team after 2^γ rows have passed since the team started playing. In each round, we divide the players of each team into groups of size 3^γ, the players compete within these groups, and there is at most one winner per group, i.e. the surviving player whose counter is highest. The winning player from each group continues to play throughout the remainder of the game, competing in further rounds. Players who are not winners withdraw from the game and do not compete in any further rounds.

At the end of the game, each team will have at most one winner. The winners from every team then compete against each other, and the player with the highest overall counter is the overall winner. Below is the general pseudocode for the initial "game" on a given stream $D = \{d_1, d_2, \ldots d_m\}$ in Algorithm 1:

Algorithm 1 The Game

1. Break the stream into rows, as described above.
2. For the i-th row, as it is read in:
   a. Sample elements from the i-th row to act as team T_i for that row.
   b. For each player t_i on team T_i:
      i. Get the initial count for that player by counting the rest of the occurrences of that element in the i-th row.
      ii. For each other team T_x with $x < i$, update the count of each player t_x on team T_x such that t_x is equal to the current element.
   c. Increment each other row's rounds-since-formation.
   d. For each other row, if 2^γ rows have passed since this row's formation:
      i. Eliminate all players whose counters are less than 2^γ
      ii. Divide that row into groups of 3^γ players.
      iii. Compete among the players in each group.
      iv. Eliminate all players from the row that did not win their groups.
3. At the end of the game compete among all remaining elements to determine the winner.

To illustrate the analysis, assume that $F_k = O(n)$ and that 1 is a heavy element that appears among every $O(n^{1-1/k})$ elements. We can make two observations. First, the counter of the player who samples 1 requires only $O(\gamma)$ bits after seeing $2^\gamma n^{1-1/k}$ elements of the stream. Also, this counter will have a nice property of linear growth: after seeing 2^γ intervals the counter will be at least 2^γ.

Second, we can observe that the sum of the frequencies of every element that is not 1 has frequency larger than λ is at most $\frac{G_k}{\lambda^{k-1}}$, where $G_k = F_k - f_1^k$.

Thus, we can bound the number of intervals with many such elements. For example, fix some constant C and let an element l be "γ-bad" if $f_l \geq 2^\gamma$ and consider an interval to be a "γ-bad" interval if it contains more than $\frac{n^{1-1/k}}{2^{C\gamma}}$ distinct bad elements. There are at most $\frac{G_k}{2^{(k-1)\gamma}} \frac{2^{C\gamma}}{n^{1-1/k}}$ such intervals.

Under the assumption that $F_k = O(n)$, and for sufficiently large k, this number is bounded by $\frac{n^{1/k}}{2^{C\gamma}}$. We conclude that the majority of the intervals (e.g 0.95%) will not be γ-bad for any γ. Let an interval that is not *bad* be called a *good* interval. Fix one such good interval nd assume, w.l.o.g, that the first player samples the heavy element, 1. In the γ-th round there are at most $3^\gamma - 1$ players that can compete with the first player. Then, because the

interval is *good* the probability to sample any element with frequency $\geq 2^\gamma$ is at most $\frac{3^\gamma}{2^{C\gamma}}$. Summing over all γ, we conclude that, with a constant probability, the heavy hitter will not compete with other players since they will expire at some earlier point in the game.

Unfortunately, the above observations are not true in general. First, the distribution of the heavy element throughout the stream can be arbitrary. For example, half of the appearances of the heavy element may occur in a single row and thus we need $\log n$ bits at the time each player starts playing the game. Second, it is possible that G_k is much larger than n in which case the number of bad intervals can be larger. It is possible that there exist intervals with the number of 1s being 2^i for every $i = 0, 1, \ldots, \lfloor \log(n) \rfloor$ and they comprise an equal percentage of the total frequency.

To overcome these problems we show that there exists a β such that there are a sufficiently large number of intervals where the number of 1s in each interval is in the range $[2^\beta, 2^{\beta+1}]$.

In general, our goal is to show that for any distribution of the heavy element in the stream there exists some β such that

1. $O\left(\frac{n^{1-2/k}}{2^{\mu\beta}}\right)$ samples from each interval are needed to sample the heavy element with a constant probability, where μ is a small constant,
2. Players that may compete with the heavy element will expire with high probability before the competition.

The space bound implies that the problem can be solved without knowledge of the value of β. To address the case when $F_k > n$ we play an additional set of games to search for a parameter α. This parameter α is used to define the number of columns in the matrix we play the game on, specifically $2^{-\alpha} n^{1-1/k}$.

2.5 A Sequence of Games

An exhaustive search of the range of α, β, which we show in the full version of the paper, [9], is at most logarithmic, and yields the sequence of games that eventually constitutes our heavy hitter algorithm. The total cost of playing all of the games is still $O(n^{1-2/k})$. As stated before, we will prove that the cost for these games is geometric and, after some slight modifications discussed in this paper, yields the desired overall cost.

Our proof of correctness will rely heavily on what we term the "Noisy Lemma" (see full version of paper, [9]). In this lemma we aim to show that at least one event in a collection of "good" events will come to pass with at least a certain probability, even if each event is impeded by a number of "noisy" events that can prevent the event from occurring. This lemma can then be applied to show that at least one player corresponding to the actual heavy hitter will win overall, even if there is a chance that other players with large but not heavy elements will win some games.

Having established this algorithm, we show we can eliminate the *log* factor associated with storing counters. This is because we store $O(\gamma^{O(1)})$ bits per counter. It remains to show that we can store the ID of each player in sufficiently small space to achieve our desired bound. In order to do this, we will transfer the duty of tracking the identity of each player from a deterministic ID to a hashed signature.

2.6 Signatures instead of IDs

Given a new element of the stream, our algorithm needs to be able to differentiate elements for the following reasons:

- If the new element has the same ID as one of the samples, then the stored counter of the sample should be incremented.

- If the new element has been chosen as a new sample for one of the players, it is necessary to compare the IDs of the new elements and the current sample. If they are the same, we increment the counter; if they are different, we have to replace the sample.

Since there are n possible elements, $\log n$ bits are required to identify all of the IDs deterministically. Note that after $O(\log \log n)$ rounds a team with initially w active players will only have $\frac{w}{\log^{\Omega(1)} n}$ active players. Thus, $O(n^{1-2/k})$ bits are sufficient to store all IDs of sampled elements in all tables for all old rows for which at least $O(\log \log n)$ rounds have passed.

Therefore, we only need to take care of the first $\log \log n$ rounds each row plays. Our goal is to reach $O(\gamma^{O(1)})$ bits per signature. To achieve this goal, we use random signatures. Unfortunately, if we simply hash $[n]$ into a range of $[2^{\gamma^{O(1)}}]$, the number of collisions per row will still be polynomial in n for small γ's.

In contrast, a small (constant) probability of collision can be shown for sets of small cardinalities. Thus, to use signatures we have to reduce the cardinality of the set. We do so by implementing a sampling procedure with an additional independent hash function. We choose the function carefully so that in a game with parameter β the probability to sample the heavy element for a row is preserved. First, we hash elements into a range $g : [n] \mapsto [t]$, where t is the number of columns in our matrix, and allow only those elements with values smaller than 2^β to be sampled. We then compute signatures only for the elements in the "pool" Γ of all elements that pass the g filter. With constant probability, $|\Gamma| = O(2^\beta)$ and no element will have the same signature as the heavy element (See the next section for more detail).

The same argument will work for any 2^γ rows if the length of the signature is $\Omega(\gamma)$. Thus, we can use γ bits to represent all of the IDs. Then, after $\log \log n$ rounds, the cardinality will be small enough such that we are able to switch our method and use the real ID of a given player's element. We implement this as follows: after $\log \log n$ rounds, we take the full ID of the first element that can be sampled and has a matching signature to the player, and assign it as the new ID of the player. We then count the frequency of elements based on this new ID. With constant probability this will be the same heavy element that we used to generate the signature to begin with.

While this technique reduces the space required, the downside is that there will be collisions for many of the w players and as a result we need to overcome two technical issues. First, due to multiple IDs being hashed to the same signature, the counters of the players can be larger than the frequency of the sampled element they are supposed to be counting. Second, if the heavy element is sampled from row i it can now be incorrectly compared with many non-heavy elements from rows $\{i, \ldots, i + 2^\gamma\}$ that collide with another value initially sampled in row i. Intuitively, this can cause the counter for a given signature to be large due to many non-heavy elements hashing to the same signature. Because much of the analysis on the correctness of the algorithm is based on the counters of players who have sampled non-heavy elements, this difference must be addressed as well. We overcome both of these problems as follows.

First, after we have progressed far enough to assign the real ID in addition to the signature, we will add a new counter. We will stop incrementing the old counter, and the new counter will count only the frequency of elements with the chosen ID. Thus, we will no longer be counting based on the signature, and we will separate the values counted by the signature from the values counted by the ID. Then, after more rounds, we will switch to using only the new counter and thus the first problem will be fixed. This change creates an additional problem: some appearances of the heavy element might be discarded. We will ensure that

the new counter will be polynomially larger than the old counter at the time when it will be discarded. Thus, the change is negligible and will not affect the correctness.

Second, we prove in the full version of the paper [9], that the probability that the counter of a non-heavy competitor will not increase by enough to alter the outcome of the game. We thus show collisions between non-heavy elements does not affect the correctness of the algorithm.

Therefore, by adding the use of hashed signatures to the way we differentiate elements that we can bound the amount of bits used to store all ID numbers and all signatures by $O(n^{1-2/k})$.

2.7 Two Level Sampling

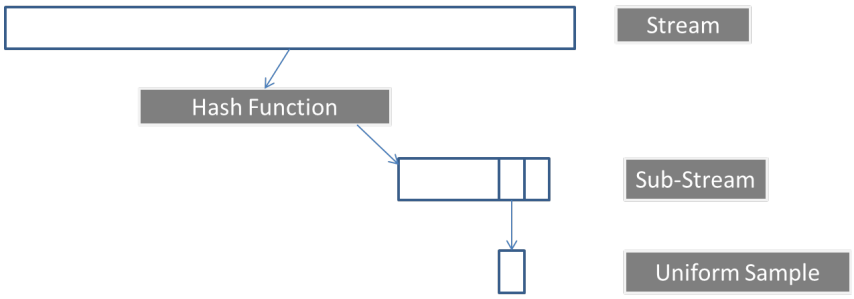


Figure 2 Two-Level Sampling.

In order to successfully use signatures instead of IDs, we also require a new method of sampling. This new method, illustrated above, combines hashing and uniform random sampling from the stream. Informally, we hash the stream and then sample uniformly from the resulting substream. Formally, the algorithm is as follows:

Algorithm 2 Two-Level Sampling(Q, λ)

1. Generate pairwise independent hash function $h : [n] \mapsto \{0, 1\}$ such that $P(h(i) = 1) = \lambda$.
2. For every element of $p \in Q$: if $h(q) = 1$ then add q to a "pool" Γ.
3. In parallel, maintain a uniform random sample L from Γ using reservoir sampling [45].
4. Return L.

2.8 An Illustrative Example

In this section we will demonstrate the main steps of our method by considering a simplified problem. Let D be a stream with the following promise: all non-zero frequencies are equal to 1 with the exception of a single element i such that the frequency of i is $f_i \geq n^{1/k}$. Furthermore, $m = \Theta(n)$ and if we split D into intervals of length $O(n^{1-1/k})$ then i appears once in each interval. Clearly, i is the heavy element and the goal of the algorithm will be to find the value of i. This simplified case is interesting because the same promise problem is used for the lower bound in [19] and in many other papers. We will thus illustrate the capability of our method by showing that a bound $O(n^{1-2/k})$ is achievable in this case.

We will assume without loss of generality that $i = 1$. This assumption does not change the analysis but simplifies our notation. In [15] it is shown that $O(n^{1-2/k})$ samples are

sufficient to solve the problem. However, each sample requires $\log n$ bits for identification (we will use a notion of "ID" to identify the value of $i \in [n]$.) As well, any known algorithm stores information about the frequency of the heavy element. This can be done by storing a sketch or an explicit approximate counter. In the most direct implementation, $\log m$ bits are required to store the counter. In this example we will assume that $\log n = \Theta(\log m)$ and we will use a single parameter $\log n$.

If $n^{1-2/k}$ independent samples are sampled from each interval then the probability to sample 1 is a constant. Next, observe that most of the time only $O(1)$ bits are needed for the counters since all frequencies except $i = 1$ are either zero or one. Thus, it is sufficient to reduce the bits for IDs.

The key idea is to replace IDs with signatures and uniform sampling with (appropriately chosen) hashing. Combining signatures of constant length with hashing ensures that the number of false positives is relatively small. Specifically, consider a hash function[4] $g : [n] \mapsto [n^{1-1/k}]$ and let the z-th sample of the i-th interval be defined as follows. Let

$$\Gamma_{i,z} = \{j : g(p_j) = z\} \tag{3}$$

where p_j are elements from the i-th interval. To obtain the final sample, we sample one element uniformly at random from $\Gamma_{i,z}$. This sampling schema is *two-level sampling* as described in the previous section. The probability that 1 is sampled using the new sampling method is still a constant. Now consider the case that each sample is represented using a signature of length $O(1)$. Suppose that we store signature SIG for the z-th sample in the i-th interval. The comparison of the sample with another element q of the stream will be defined by the following procedure. We say that they are equal if $g(q) = z$ and the signature of q is equal to SIG. Consider the case when we sample the heavy element. In this case the consecutive appearances of 1 will always be declared equal to the sample. Then, consider another case when l has been sampled and when $f_l = 1$. The probability that there will be any collision in the next interval is at most $2^{-|SIG|}$. Therefore we can exploit the probability gap between these two cases.

Specifically, deleting samples with a small number of collisions allows for increasing signatures for the remaining samples in the future intervals. After 2^γ intervals, it is possible to increase the signature by $O(1)$ bits for $\gamma = 1, 2, \ldots$. In the full version of the paper, [9], we show that the heavy element will never be discarded and that the number of active samples decreases exponentially with γ. Thus, the total expected space for storing the data is $O\left(\frac{n^{1-2/k}\gamma}{2^{\Omega(\gamma)}}\right)$. The aforementioned procedure is called the γ-th round for the i-th interval. At any moment there are at most 2^γ intervals in the γ-th round and the total space is $O(n^{1-2/k})$. For $\gamma = \Omega(\log \log n)$ storing IDs instead of signatures implies that if the heavy element is not discarded then the correct answer is produced. The algorithm works in one pass and uses $O(n^{1-2/k})$ bits.

2.9 Martingale Sketches

Having established a streaming algorithm which can efficiently compute the heavy hitters of a stream, we present a reduction of the problem of frequency moment approximation to that of finding heavy hitters. In general, this analysis will show that the problem of approximating the sum of am implicit vector is the same problem as finding the heavy

[4] It is possible to show that g can be pairwise independent.

elements of that vector, up to a constant factor. As a direct corollary, and using our new heavy hitter algorithm, we obtain a new lowest bound on space required for this problem.

The intuition behind this method is as follows: consider a vector where the sum of its elements cannot be computed directly. If the elements of the vector vary in magnitude, then some elements will have a larger impact on the sum than others. Now consider a second vector made from including or excluding each element of the first by the repeated flip of a fair coin, and then doubling the value of every included element. The expected difference between the sums of the two vectors is 0. But because of the disproportionate contribution of heavy hitters, the actual difference will most likely not be 0. If we can find the heavy hitters of the first vector, we can examine which ones were included and which were excluded in the second vector. Intuitively, the excluded ones will increase the difference between the vector sums, while the included heavy hitters will decrease it (because of the scaling up by a factor of 2, and their already large contribution to the total sum). This allows us to approximate the difference between the two vector sums. If we repeat this process for the second vector and a new vector made from including or excluding each of its elements (with the included elements having their values doubled), and so on, then the repeated differences along with the sum of the final vector can be used together to accurately approximate the sum of the first vector. Thus, finding a frequency moment is reducible to finding the heavy hitters of a series of vectors.

While the overall idea of reducing a vector sum to its heavy hitters is not new, what our algorithm provides is a cost function that is geometric by the nature of the given reduction. Thus, the total space cost required for these computations matches the lower bound for frequency computation, up to a constant factor.

3 Putting It All Together

At this point, we have provided an algorithm that finds heavy hitters in a stream which conforms to certain assumptions. Due to the lack of space, we omit many technical details from the main body of the paper in this abstract. In the full version of this paper,[9], we provide detailed analysis showing that with only several small adjustments the slightly modified version of our original algorithm will work on general streams. With these slight modifications, the technical details of which are included in the appendix, we have succeeded in providing an algorithm which proves Theorem 3.

We also explain how the first and third passes can be removed, and that we can get the algorithm to work for any $k > 3$. Consequently, the reduction proven in the Martingale Sketches section coupled with the AHE algorithm proves our main result, Theorem 8, by providing an algorithm that computes frequency moments with $k > 3$ using $O(n^{1-2/k})$ bits of memory.

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 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.

3 Alexandr Andoni, Huy L. Nguyen, Yury Polyanskiy, and Yihong Wu. Tight lower bound for linear sketches of moments. In *Proceedings of the 40th International Conference on*

Automata, Languages, and Programming – Volume Part I, ICALP'13, pages 25–32, Berlin, Heidelberg, 2013. Springer-Verlag.

4 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. In *Proceedings of the 43rd Symposium on Foundations of Computer Science*, FOCS'02, pages 209–218, Washington, DC, USA, 2002. IEEE Computer Society.

5 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, D. Sivakumar, and Luca Trevisan. Counting distinct elements in a data stream. In *Proceedings of the 6th International Workshop on Randomization and Approximation Techniques*, RANDOM'02, pages 1–10, London, UK, UK, 2002. Springer-Verlag.

6 Paul Beame, T. S. Jayram, and Atri Rudra. Lower bounds for randomized read/write stream algorithms. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, STOC'07, pages 689–698, New York, NY, USA, 2007. ACM.

7 Lakshminath Bhuvanagiri, Sumit Ganguly, Deepanjan Kesh, and Chandan Saha. Simpler algorithm for estimating frequency moments of data streams. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, SODA'06, pages 708–713, New York, NY, USA, 2006. ACM.

8 Vladimir Braverman, Ran Gelles, and Rafail Ostrovsky. How to catch l 2-heavy-hitters on sliding windows. In Ding-Zhu Du and Guochuan Zhang, editors, *Computing and Combinatorics*, volume 7936 of *Lecture Notes in Computer Science*, pages 638–650. Springer Berlin Heidelberg, 2013.

9 Vladimir Braverman, Jonathan Katzman, Charles Seidell, and Gregory Vorsanger. Approximating Large Frequency Moments with $O(n^{1-2/k})$ Bits. *CoRR*, abs/1401.1763, 2014.

10 Vladimir Braverman and Rafail Ostrovsky. Smooth histograms for sliding windows. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'07, pages 283–293, Washington, DC, USA, 2007. IEEE Computer Society.

11 Vladimir Braverman and Rafail Ostrovsky. Effective computations on sliding windows. *SIAM J. Comput.*, 39(6):2113–2131, March 2010.

12 Vladimir Braverman and Rafail Ostrovsky. Measuring independence of datasets. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC'10, pages 271–280, New York, NY, USA, 2010. ACM.

13 Vladimir Braverman and Rafail Ostrovsky. Recursive sketching for frequency moments. *CoRR*, abs/1011.2571, 2010.

14 Vladimir Braverman and Rafail Ostrovsky. Zero-one frequency laws. In *Proceedings of the 42nd ACM symposium on Theory of computing*, STOC'10, pages 281–290, New York, NY, USA, 2010. ACM.

15 Vladimir Braverman and Rafail Ostrovsky. Approximating large frequency moments with pick-and-drop sampling. Accepted to the 16th. International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'2013)., 2013.

16 Vladimir Braverman and Rafail Ostrovsky. Generalizing the layering method of Indyk and Woodruff: Recursive sketches for frequency-based vectors on streams. Accepted to the 16th. International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'2013)., 2013.

17 Vladimir Braverman, Rafail Ostrovsky, and Carlo Zaniolo. Optimal sampling from sliding windows. *J. Comput. Syst. Sci.*, 78(1):260–272, January 2012.

18 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. Robust lower bounds for communication and stream computation. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, STOC'08, pages 641–650, New York, NY, USA, 2008. ACM.

19 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *IEEE Conference on Computational Complexity*, pages 107–117, 2003.

20 Don Coppersmith and Ravi Kumar. An improved data stream algorithm for frequency moments. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA'04, pages 151–156, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

21 Graham Cormode. Continuous distributed monitoring: a short survey. In *Proceedings of the First International Workshop on Algorithms and Models for Distributed Event Processing*, AlMoDEP'11, pages 1–10, New York, NY, USA, 2011. ACM.

22 Graham Cormode, Mayur Datar, Piotr Indyk, and S. Muthukrishnan. Comparing data streams using hamming norms (how to zero in). *IEEE Trans. on Knowl. and Data Eng.*, 15(3):529–540, 2003.

23 Mayur Datar, Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Maintaining stream statistics over sliding windows. *SIAM J. Comput.*, 31(6):1794–1813, 2002.

24 J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. An approximate l1-difference algorithm for massive data streams. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, FOCS'99, pages 501–, Washington, DC, USA, 1999. IEEE Computer Society.

25 Philippe Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, September 1985.

26 Sumit Ganguly. Estimating frequency moments of data streams using random linear combinations. In *APPROX-RANDOM*, pages 369–380, 2004.

27 Sumit Ganguly. Polynomial estimators for high frequency moments. *CoRR*, abs/1104.4552, 2011.

28 Sumit Ganguly. A lower bound for estimating high moments of a data stream. *CoRR*, abs/1201.0253, 2012.

29 Sumit Ganguly and Graham Cormode. On estimating frequency moments of data streams. In *Proceedings of the 10th International Workshop on Approximation and the 11th International Workshop on Randomization, and Combinatorial Optimization. Algorithms and Techniques*, APPROX'07/RANDOM'07, pages 479–493, Berlin, Heidelberg, 2007. Springer-Verlag.

30 Phillip B. Gibbons and Srikanta Tirthapura. Distributed streams algorithms for sliding windows. In *Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures*, SPAA'02, pages 63–72, New York, NY, USA, 2002. ACM.

31 Andre Gronemeier. Asymptotically optimal lower bounds on the nih-multi-party information. *CoRR*, abs/0902.1609, 2009.

32 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006.

33 Piotr Indyk and David Woodruff. Optimal approximations of the frequency moments of data streams. In *STOC'05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 202–208, New York, NY, USA, 2005. ACM.

34 T. S. Jayram, Andrew McGregor, S. Muthukrishnan, and Erik Vee. Estimating statistical aggregates on probabilistic data streams. In *PODS'07: Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 243–252, New York, NY, USA, 2007. ACM.

35 T. S. Jayram and David Woodruff. Optimal bounds for johnson-lindenstrauss transforms and streaming problems with sub-constant error. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'11, pages 1–10. SIAM, 2011.

36 T.S. Jayram. Hellinger strikes back: A note on the multi-party information complexity of and. In Irit Dinur, Klaus Jansen, Joseph Naor, and José Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 5687 of *Lecture Notes in Computer Science*, pages 562–573. Springer Berlin Heidelberg, 2009.

37 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*, SODA'10, pages 1161–1178, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

38 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'10, pages 41–52, New York, NY, USA, 2010. ACM.

39 Ping Li. Compressed counting. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'09, pages 412–421, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

40 Yi Li and DavidP. Woodruff. A tight lower bound for high frequency moment estimation with small error. In Prasad Raghavendra, Sofya Raskhodnikova, Klaus Jansen, and JoséD.P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 8096 of *Lecture Notes in Computer Science*, pages 623–638. Springer Berlin Heidelberg, 2013.

41 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.

42 S. Muthukrishnan. Data streams: algorithms and applications. *Found. Trends Theor. Comput. Sci.*, 1(2):117–236, 2005.

43 Jelani Nelson and David P. Woodruff. Fast Manhattan sketches in data streams. In *PODS'10: Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems of data*, pages 99–110, New York, NY, USA, 2010. ACM.

44 Eric Price and David P. Woodruff. Applications of the shannon-hartley theorem to data streams and sparse recovery. In *ISIT*, pages 2446–2450, 2012.

45 J. S. Vitter. *Random sampling with a reservoir*. ACM Transactions on Mathematical Software, v.11 n.1, pp.37–57, 1985.

46 David Woodruff. Optimal space lower bounds for all frequency moments. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA'04, pages 167–175, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

47 David P. Woodruff. Frequency moments. In *Encyclopedia of Database Systems*, pages 1169–1170. Springer, 2009.

48 David P. Woodruff and Qin Zhang. Tight bounds for distributed functional monitoring. In *Proceedings of the 44th symposium on Theory of Computing*, STOC'12, pages 941–960, New York, NY, USA, 2012. ACM.

Certifying Equality With Limited Interaction*

Joshua Brody[1], Amit Chakrabarti[2], Ranganath Kondapally[2], David P. Woodruff[3], and Grigory Yaroslavtsev[4]

1 **Swarthmore College, Swarthmore, PA, USA**
joshua.e.brody@gmail.com
2 **Dartmouth College, Dartmouth, NH, USA**
ac@cs.dartmouth.edu, ranganathk@gmail.com
3 **IBM Almaden, Almaden, CA, USA**
dpwoodru@us.ibm.com
4 **Brown University, ICERM, Providence, RI, USA**
grigory@grigory.us

Abstract

The EQUALITY problem is usually one's first encounter with communication complexity and is one of the most fundamental problems in the field. Although its deterministic and randomized communication complexity were settled decades ago, we find several new things to say about the problem by focusing on three subtle aspects. The first is to consider the *expected* communication cost (at a worst-case input) for a protocol that uses limited interaction—i. e., a bounded number of rounds of communication—and whose error probability is zero or close to it. The second is to treat the *false negative* error rate separately from the *false positive* error rate. The third is to consider the *information cost* of such protocols. We obtain asymptotically optimal rounds-versus-cost tradeoffs for EQUALITY: both expected communication cost and information cost scale as $\Theta(\log\log\cdots\log n)$, with $r-1$ logs, where r is the number of rounds. These bounds hold even when the false negative rate approaches 1. For the case of zero-error communication cost, we obtain essentially matching bounds, up to a tiny additive constant. We also provide some applications.

1998 ACM Subject Classification F.1.2 Modes of Computation

Keywords and phrases equality, communication complexity, information complexity

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.545

1 Introduction

1.1 Context

Over the last three decades, communication complexity [44] has proved itself to be among the most useful of abstractions in computer science. A number of basic problems in communication complexity have found a wide range of applications throughout the theory of computing, with EQUALITY, INDEX, and DISJOINTNESS being notable superstars.

Revisiting these basic problems and asking more nuanced questions or studying natural variants has extended their range of application. We highlight two examples. Our first example is DISJOINTNESS. The optimal $\Omega(n)$ lower bound for this problem [29, 41] was already considered one of the major results in communication complexity before DISJOINTNESS was revisited in the *multi-party* number-in-hand model to obtain a number of data stream lower

* Grigory Yaroslavtsev was supported by an Institute Postdoctoral Fellowship at Brown ICERM.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 545–581

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

bounds [3, 4, 13, 23] culminating in optimal space bounds for the (higher) frequency moments. Later, DISJOINTNESS was revisited in an *asymmetric* communication setting [40] yielding an impressive array of lower bounds for data structures in the cell-probe model. Very recently, DISJOINTNESS was revisited yet again in a *high-error* setting, yielding deep insights into extended formulations for the MAX-CLIQUE problem [8]. Our second example is INDEX. The straightforward $\Omega(n)$ lower bound on its one-way communication complexity [1] is already an important starting point for numerous other lower bounds. Revisiting INDEX in an *interactive* communication setting and considering communication tradeoffs has led to new classes of data stream lower bounds for memory-checking problems [34, 12, 14]. Separately, lower bounding the *quantum* communication complexity of INDEX [39] has led, among other things, to strong lower bounds for locally decodable codes [30, 16].

1.2 Our Results

In this work we revisit the EQUALITY problem: Alice and Bob each hold an n-bit string, and their task is to decide whether these strings are equal. This is arguably the most basic communication problem that admits a nontrivial protocol: using randomization and allowing a constant error rate, the problem can be solved with just $O(1)$ communication (this becomes $O(\log n)$ if one insists on private coins only); see, e. g., Kushilevitz and Nisan [32, Example 3.13] and Freivalds [22]. This is why a student's first encounter with communication complexity is usually through the EQUALITY problem. Such a fundamental problem deserves the most thorough of studies.

At first glance, EQUALITY might appear "solved": its deterministic communication complexity is at least n, whereas its randomized complexity is $O(1)$ as noted above, as is its *information complexity* [6] (for more on this, see Section 1.3). However, one can ask the following more nuanced question. What happens if Alice and Bob want to be *certain* (or nearly certain) that their inputs are indeed equal when the protocol directs them to say so? And what happens if Alice and Bob want to run a protocol with limited interaction, i. e., a bounded number of back-and-forth rounds of communication?

Formally, let $\text{EQ}_n : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ denote the Boolean function that underlies this communication problem, defined by $\text{EQ}_n(x,y) = 1 \iff x = y$. Consider the zero-error case: the players must always correctly output $\text{EQ}_n(x,y)$ on every input (x,y). However, the players may use a randomized protocol and their goal is to minimize the *expected* number of bits they exchange. If their protocol is required to use only one round—this means that Alice sends a message to Bob, who then outputs the answer—then it is easy to see that Alice's message must uniquely identify her input to Bob. From this it is easy to show that on some input, x, Alice must send at least n bits to Bob, even in expectation.

Things improve a lot if one allows two rounds of communication—Alice sends a message to Bob, who replies to Alice, who then outputs the answer. Using standard techniques, Alice can send Bob a $\lceil \log n \rceil$-bit[1] *fingerprint* of x. When $x \neq y$, this fingerprint demonstrates with probability at least $1 - 1/n$ that $\text{EQ}_n(x,y) = 0$. If necessary, Bob responds to this failure by sending y to Alice, which costs only 1 bit in expectation. The net result is an expected communication cost of $O(\log n)$ on unequal inputs, and $O(n)$ on equal inputs. Generalizing this idea, we obtain an r-round protocol where the expected cost drops to $O(\text{ilog}^{r-1} n)$ on unequal inputs, where $\text{ilog}^j n := \log\log\cdots\log n$ (with j logs).

Our main high-level message in this work is that *the above tradeoff between the number of*

[1] Throughout this paper we use "log" to denote the logarithm to the base 2.

rounds and the communication cost is optimal, and that this remains the case even allowing for some false positives, even allowing for a false negative rate of $1 - o(1)$, and *even if we consider information cost*. We shall get precise about information cost measures in Section 2, but for now we remark that an information cost lower bound is stronger than a communication cost bound, even in our expected-cost model.

While our main focus is on EQUALITY, our rounds-versus-information tradeoff can be applied to three other problems: OR-EQUALITY, DISJOINTNESS, and PRIVATE-INTERSECTION. It is well known that information cost has clean direct-sum properties [15, 4, 5]. Together with our results for EQUALITY, this gives us lower bounds for the bounded-round randomized communication complexity of the OR-EQUALITY problem, whose underlying function is $\text{OREQ}_{n,k} : \{0,1\}^{nk} \times \{0,1\}^{nk} \to \{0,1\}$, defined by $\text{OREQ}_{n,k}(x_1, \ldots, x_k, y_1, \ldots, y_k) = \bigvee_{i=1}^{k} \text{EQ}_n(x_i, y_i)$: Alice holds each $x_i \in \{0,1\}^n$ and Bob holds each $y_i \in \{0,1\}^n$. The OREQ problem is closely related to DISJOINTNESS, especially the variant called *small set disjointness* or k-DISJ_N. Here, Alice and Bob are given sets $A, B \subseteq \{1, 2, \ldots, N\}$ respectively, with the promise that $|A| \leq k$ and $|B| \leq k$, where $1 \leq k \leq N$. Their goal is to output 1 iff $A \cap B = \varnothing$. Using this close relation (see Lemma 54 for a formal treatment), we obtain bounded-round lower bounds for k-DISJ as well. For low-error protocols, our bounds asymptotically match those recently given by Sağlam and Tardos [42]; our proof is quite different and should be of independent interest. Additionally, our lower bound also allows a false negative rate of $1 - o(1)$ — we show that if players only need to verify equal inputs with probability, say, 0.001, the problem remains difficult.

Another key property is that information cost is a measure of *privacy* of a protocol for a function f. Klauck [31] defines[2] the privacy of a protocol Π with respect to a distribution μ:

$$\text{PRIV}^{\mu}(\Pi) := \text{I}(X : \Pi(X,Y) \mid Y, f(X,Y)) + \text{I}(Y : \Pi(X,Y) \mid X, f(X,Y)) \,.$$

This definition coincides with $\text{IC}_{\mu}(\Pi)$ up to the conditioning on $f(X,Y)$ in the mutual information expressions. However, in many cases, including this paper, this conditioning does not asymptotically affect the definition, and one has $\text{PRIV}^{\mu}(\Pi) = \Theta(\text{icost}^{\mu}(\Pi))$. One can then naturally define $\text{PRIV}_{\delta}(f) = \min_{\delta\text{-error } \Pi} \max_{\text{input dist } \mu} \text{PRIV}^{\mu}(\Pi)$, and one has that $\text{PRIV}_{\delta}(f) = \Theta(\text{IC}_{\delta}(f))$.

There is a large body of work on trying to solve EQUALITY privately. These are known as *private equality tests* in the cryptography and privacy literature [19, 38]. A harder problem is that of determining the intersection $A \cap B$ of sets A, B in some finite universe, where each of $|A|$ and $|B|$ is promised to be at most k. This is a fundamental problem studied in private datamining, see, e. g., the work by Freedman et al. [21]. We refer to the latter problem as the PRIVATE-INTERSECTION problem. It is worth noting that the PRIVATE-INTERSECTION problem is studied both under computational assumptions on the players, as in the work by Freedman et al. [21], and also using information-theoretic notions of privacy, such as $\text{PRIV}_{\delta}(f)$, as in the work by Ada et al. [2]. Note that for the PRIVATE-INTERSECTION problem, we are asking for a correct protocol which reveals as little information about A and B as possible, with no constraints on the communication.

While the information complexity of PRIVATE-INTERSECTION is known to be $\Theta(k)$, in certain applications the players can only exchange messages in a bounded number r of rounds, since, e. g., the number of rounds is related to the overall latency of the protocol. The number of rounds may in fact influence the latency drastically while the actual number

[2] We have replaced the max in Klauck's definition with a sum; this agrees with Klauck's original definition up to a factor of 2.

of bits communicated may not. This is because the more interactive protocols are, i. e., the larger the number of rounds, the more coordination is needed between the players, which may not be possible if, e. g., a player goes offline.

We apply our information complexity lower bound for EQUALITY to the PRIVATE-INTERSECTION problem in which each player should locally output the entire set intersection $S \cap T$. Our information complexity lower bound for EQUALITY can be combined with a recent direct sum theorem with *aborts*, which (roughly speaking) states that the information complexity of solving all k copies of a problem is k times the information cost of solving each copy with a protocol that is allowed to output "abort" with a constant $1/10$ probability but, conditioned on non-abortion, is correct with a very high $1 - 1/k$ probability [37].[3] By changing such a protocol for EQUALITY so that whenever it would have output "abort", it instead declares that $x \neq y$, we show how to obtain an $\Omega(k \operatorname{ilog}^r k)$ information cost bound for PRIVATE-INTERSECTION for any r-round protocol with constant success probability. As $\mathrm{I}(\Pi : S \mid T, S \cap T) + \mathrm{I}(\Pi : T \mid S, S \cap T) = \mathrm{I}(\Pi : S \mid T) + \mathrm{I}(\Pi : T \mid S) \pm O(k)$, it follows that $\mathrm{PRIV}_{1/3}(\text{PRIVATE-INTERSECTION}) = \Omega(k \operatorname{ilog}^r k)$.

For a concise (yet technically precise) listing of our results, please see Section 2.

1.3 Related Work

The study of the EQUALITY problem goes back to the original communication complexity paper of Yao [44], who showed that the deterministic communication complexity of EQ_n is at least n, using a fooling set argument. Mehlhorn and Schmidt [35] developed the *rank lower bound* technique, which can recover this result. They further examined OR-EQUALITY, giving a lower bound of nk bits for deterministic protocols that compute $\mathrm{OREQ}_{n,k}$ via the rank technique. They also gave $O(n + \log n)$ and $O(n \log n)$ bounds for the nondeterministic and co-nondeterministic communication complexities of $\mathrm{OREQ}_{n,n}$, respectively. Furthermore, they studied the "Las Vegas" communication complexity of $\mathrm{OREQ}_{n,n}$, which brought them close to some of the things we study here. Specifically, they gave a zero-error private-coin randomized protocol such that the expected communication cost on any inputs $(x_1, \ldots, x_n, y_1, \ldots, y_n)$ is at most $O(n(\log n)^2)$.

Feder et al. [20] studied the randomized communication complexity of EQUALITY in the direct-sum setting. Here, players have k strings each and must compute $(\mathrm{EQ}_n(x_1, y_1), \ldots, \mathrm{EQ}_n(x_k, y_k))$: thus, the output is a k-bit string. Feder et al. showed that $O(k)$ communication suffices to compute EQUALITY on all k instances, with error *exponentially* small in k. This shows that the "amortized" communication complexity of EQ_n is $O(1)$, even under tiny error. More recently, Braverman and Rao [9] showed that amortized communication complexity nearly equals *information* complexity. Furthermore, Braverman [6] gave a specific protocol for EQ_n that has zero error and achieves information cost $O(1)$ regardless of the input distribution.

---

[3] It is crucial for us to use a strong direct sum theorem of [37] in the lower bound for PRIVATE-INTERSECTION. Unlike generic direct sum and direct product theorems which apply to any function the strong direct sum of [37] only applies to EQUALITY-type functions but gives a much stronger guarantee in the constant error regime that we study here. This is in contrast with the bounded round direct product theorem of [26, 27] (and other similar results such as [28]), who show that for r-round public-coin randomized information complexity $\mathrm{IC}^{r,\mathrm{pub}}_{1-(1-\varepsilon/2)^{\Omega(k\varepsilon^2/r^2)}}(f^k) = \Omega\big(\ (\varepsilon k/r) \cdot (IC^{r,\mathrm{pub}}_{\varepsilon}(f) - O(r^2/\varepsilon^2))\ \big)$, where $\varepsilon > 0$ is arbitrary (the results of [26, 27] are stated in terms of communication complexity but their techniques also imply an information cost lower bound). One cannot apply this theorem to our problem, as one would need to set $\varepsilon = \Theta(k^{-1/3})$ to obtain our results. Because $\mathrm{IC}^{r,\mathrm{pub}}_{1/k^{1/3}}(\text{EQUALITY}) = o(k^{2/3})$ this theorem gives a trivial bound.

The problem $\text{OREQ}_{n,k}$ is potentially easier than the k-fold direct sum of EQ_n, and has itself been studied a few times before. Chakrabarti et al. [15] showed that its simultaneous-message complexity is $\Omega(k\sqrt{n})$, which is k times the complexity of EQ_n in that model. More recently, Kushilevitz and Weinreb [33] studied the deterministic complexity of $\text{OREQ}_{n,k}$ under the promise that $x_i = y_i$ for at most one $i \in [k]$. Computing $\text{OREQ}_{n,k}$ under this "0/1 intersection" promise is closely related to the clique-vs.-independent set problem. In this problem, Alice is given a clique in a graph. Bob is given an independent set, and they must decide if their inputs intersect. Kushilevitz and Weinreb were able to show that computing $\text{OREQ}_{n,k}$ under this promise still requires $\Omega(kn)$ communication whenever $k \leq n/\log n$. Extending this lower bound to the setting where $k = n$ is an important open problem with several implications.

For the k-DISJ problem, Håstad and Wigderson [25] gave an $O(k)$-bit randomized protocol; a matching lower bound follows easily from the $\Omega(n)$ lower bound for general DISJOINTNESS. The Håstad–Wigderson protocol is clever and crucially exploits both the public randomness and the interactive communication between players. Sağlam and Tardos [42] extend this protocol to interpolate between the one-round and unbounded-round situations, showing that to compute k-DISJ in r rounds, $\Theta(k \operatorname{ilog}^r k)$ bits are necessary and sufficient. This lower bound extends tight $\Omega(k \log k)$ lower bounds for one-round protocols recently given by Dasgupta, Kumar, and Sivakumar [18] and by Buhrman et al. [11]

Since initial announcement of this work [10], we have learned that the communication complexity lower bound of Sağlam and Tardos [42], together with work of Harsha et al. [24] also give lower bounds for information complexity of OR-EQUALITY and similarly DISJOINTNESS. Additionally, with the recent direct product theorem for bounded-round communication complexity of Jain et al. [27] and the existing result equating information and amortized communication of Braverman and Rao [9] these results also extend to give information complexity lower bounds for bounded-round protocols for EQUALITY. Still, EQUALITY is one of the most important communication complexity problems; as such, it deserves careful study. Our information cost lower bounds are more direct and shed more light on this important problem. Additionally, previous results do not differentiate between errors for false positives and false negatives and cannot therefore admit the high false negative rate our bounds apply to.

The recent work of Braverman et al. [7] is similar in spirit to some of our results. They consider zero-error communication protocols for the even more fundamental AND function, obtaining exact information cost bounds. From this they derive nearly exact communication bounds for low-error protocols for DISJOINTNESS and k-DISJ. They also consider rounds-vs.-information tradeoffs for AND, showing that the information complexity of r-round protocols decays as $\Theta(1/r^2)$. Our work shows that the information complexity of EQUALITY decays exponentially with each additional round.

1.4 Road Map

The rest of the paper is organized as follows. Section 2 gives careful definitions of our model of computation and error and cost measures, followed by a listing of all our results. The listing provides pointers to later sections of the paper where these results are proved. Section 3 provides a sketch of our main result, which gives an information cost lower bound for EQUALITY.

We include complete details of our results including full proofs in the Appendix. Section A gives basic definitions and lemmas relating to information theory. The next two sections provide some warm-up. Section B gives upper bounds for EQUALITY including the iterated-log upper bound described informally above. Section C gives matching lower bounds for expected

communication cost, first under zero error and then under two-sided error. Though the proofs in Sections B and C are not too complex, the combined story they tell is important. Together, these results paint a nearly complete picture of the behavior of EQUALITY in a bounded-round expected-communication setting, and highlight the importance of studying YES and NO instances separately.

Section D contains the full proof of our Main Theorem, which gives an information cost lower bound for EQUALITY. Section E obtains lower bounds for OREQ and k-DISJ as an application of the Main Theorem. Finally, Section F obtains lower bounds for PRIVATE-INTERSECTION.

2 Definitions and Formal Statement of Results

Throughout this paper we reserve the symbols "n" for input length of EQUALITY instances, "k" for list length of OR-EQUALITY instances and set size of k-DISJ instances, and "N" for universe size of k-DISJ instances. Many definitions and results will be parametrized by these quantities but to keep the notation clean we shall not make this parametrization explicit. We tacitly assume that n, k and N are sufficiently large integers.

Unless otherwise stated, all communication protocols appearing in this paper are public-coin randomized protocols involving two players named Alice and Bob. Because our work concerns expected communication cost in a bounded-round setting, we make the following careful definition of what communication is allowed. In each round, the player whose turn it is to speak sends the other player a message from a *prefix-free* subset[4] of $\{0,1\}^*$. This subset can depend on the communication history. After the final round in the protocol, the player that receives the last message announces the output (which, for us, is always a single bit): this announcement does not count as a round.

Let $\mathcal{P}$ be a communication protocol that takes inputs $(x, y) \in \mathcal{X} \times \mathcal{Y}$. The *transcript* of $\mathcal{P}$ on input (x, y) is defined to be the concatenation of the messages sent by the players, in order, as they execute $\mathcal{P}$ on (x, y). We denote this transcript by $\mathcal{P}(x, y)$ and remark that it is, in general, a random variable. We include the output as the final "message" in the transcript. We denote the output of a transcript $\mathtt{t}$ by $\mathrm{out}(\mathtt{t})$. We denote the length of a binary string z by $|z|$. The *communication cost* and *worst-case communication cost* of $\mathcal{P}$ on input (x, y) are defined to be

$$\mathrm{cost}(\mathcal{P}; x, y) := \mathbb{E}\left[|\mathcal{P}(x, y)|\right], \quad \text{and} \quad \mathrm{cost}^*(\mathcal{P}; x, y) := \max |\mathcal{P}(x, y)|,$$

where the expectation and the max are taken over the protocol's random coin tosses.

We now define complexity measures based on this notion of communication cost. Ordinarily we would just define the communication complexity of a function f as the minimum over protocols for f of the worst-case (over all inputs) cost of the protocol. When $f = \mathrm{EQ}_n$, such a measure turns out to be too punishing, and hides the subtleties that we seek to study. Notice that the r-round protocol outlined in Section 1.2 achieves its cost savings only on unequal inputs, i. e., on $f^{-1}(0)$. On inputs in $f^{-1}(1)$, the protocol ends up costing at least n bits. The intuition is that it is much cheaper for Alice and Bob to *refute* the purported equality of their inputs than to *verify* it. Indeed, verification is so hard that interaction has no effect on the verification cost, whereas each additional round of communication decreases refutation cost exponentially.

[4] A set of strings is said to be prefix-free if no string in the set is a proper prefix of any other.

In fact, this intuition can be turned into precise theorems, both in zero-error and positive-error settings, as we shall see. To formalize things, we now define a family of complexity measures.

▶ **Definition 1** (Cost, Error, and Complexity Measures)**.** Let $\mathcal{P}$ be a protocol that is supposed to compute a Boolean function $f : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$. We define its *refutation cost*, *verification cost*, *overall cost*, *refutation error* (or false positive rate, or soundness error), and *verification error* (or false negative rate, or completeness error) as follows, respectively:

$$\begin{aligned}
\mathrm{rcost}(\mathcal{P}) &:= \max\nolimits_{(x,y)\in f^{-1}(0)} \mathrm{cost}(\mathcal{P}; x, y)\,,\\
\mathrm{vcost}(\mathcal{P}) &:= \max\nolimits_{(x,y)\in f^{-1}(1)} \mathrm{cost}(\mathcal{P}; x, y)\,,\\
\mathrm{cost}(\mathcal{P}) &:= \max\nolimits_{(x,y)\in \mathcal{X}\times\mathcal{Y}} \mathrm{cost}(\mathcal{P}; x, y)\,,\\
\mathrm{rerr}(\mathcal{P}) &:= \max\nolimits_{(x,y)\in f^{-1}(0)} \Pr[\mathrm{out}(\mathcal{P}(x,y)) = 1]\,,\\
\mathrm{verr}(\mathcal{P}) &:= \max\nolimits_{(x,y)\in f^{-1}(1)} \Pr[\mathrm{out}(\mathcal{P}(x,y)) = 0]\,.
\end{aligned}$$

Let λ be a probability distribution on the input space $\mathcal{X} \times \mathcal{Y}$. We then define the λ-distributional error $\mathrm{err}^{\lambda}(\mathcal{P})$ as well as the λ-distributional refutation cost, etc., as follows:

$$\begin{aligned}
\mathrm{rcost}^{\lambda}(\mathcal{P}) &:= \mathbb{E}_{(X,Y)\sim\lambda}[\mathrm{cost}(\mathcal{P}; X, Y) \mid f(X,Y) = 0]\,,\\
\mathrm{vcost}^{\lambda}(\mathcal{P}) &:= \mathbb{E}_{(X,Y)\sim\lambda}[\mathrm{cost}(\mathcal{P}; X, Y) \mid f(X,Y) = 1]\,,\\
\mathrm{cost}^{\lambda}(\mathcal{P}) &:= \mathbb{E}_{(X,Y)\sim\lambda}[\mathrm{cost}(\mathcal{P}; X, Y)]\,,\\
\mathrm{rerr}^{\lambda}(\mathcal{P}) &:= \mathbb{E}_{(X,Y)\sim\lambda}[\Pr[\mathrm{out}(\mathcal{P}(X,Y)) = 1 \mid f(X,Y) = 0]]\,,\\
\mathrm{verr}^{\lambda}(\mathcal{P}) &:= \mathbb{E}_{(X,Y)\sim\lambda}[\Pr[\mathrm{out}(\mathcal{P}(X,Y)) = 0 \mid f(X,Y) = 1]]\,,\\
\mathrm{err}^{\lambda}(\mathcal{P}) &:= \mathbb{E}_{(X,Y)\sim\lambda}[\Pr[\mathrm{out}(\mathcal{P}(X,Y)) \neq f(X,Y)]\,.
\end{aligned}$$

We shall usually restrict $\mathcal{P}$ to be deterministic when considering these distributional measures. Although these measures depend on both $\mathcal{P}$ and f, we do not indicate f in our notation to keep things simple.

Let $r \geq 1$ be an integer and let $\varepsilon, \delta \in [0,1]$ be reals. We define the r-round randomized *refutation complexity* and r-round λ-distributional refutation complexity of f as follows, respectively:

$$\begin{aligned}
\mathrm{R}^{(r),\mathrm{ref}}_{\varepsilon,\delta}(f) &:= \min\{\mathrm{rcost}(\mathcal{P}) :\ \mathcal{P} \text{ uses } r \text{ rounds, } \mathrm{rerr}(\mathcal{P}) \leq \varepsilon, \mathrm{verr}(\mathcal{P}) \leq \delta\}\,,\\
\mathrm{D}^{\lambda,(r),\mathrm{ref}}_{\varepsilon,\delta}(f) &:= \min\{\mathrm{rcost}^{\lambda}(\mathcal{P}) :\ \mathcal{P} \text{ is deterministic and uses } r \text{ rounds, } \mathrm{rerr}^{\lambda}(\mathcal{P}) \leq \varepsilon,\\
&\qquad\qquad \mathrm{verr}^{\lambda}(\mathcal{P}) \leq \delta\}\,.
\end{aligned}$$

We also define measures of *verification complexity* and *overall complexity* analogously, replacing "rcost" above with "vcost" and "cost" respectively, and denote them by

$$\mathrm{R}^{(r),\mathrm{ver}}_{\varepsilon,\delta}(f)\,,\ \mathrm{D}^{\lambda,(r),\mathrm{ver}}_{\varepsilon,\delta}(f)\,,\ \mathrm{R}^{(r)}_{\varepsilon,\delta}(f)\,,\ \text{and}\ \mathrm{D}^{\lambda,(r)}_{\varepsilon,\delta}(f)\,,$$

respectively. We define the *total complexity* of f as follows:

$$\begin{aligned}
\mathrm{R}^{*,(r)}_{\varepsilon,\delta}(f) &:= \min\{\mathrm{cost}^{*}(\mathcal{P}) :\ \mathcal{P} \text{ uses } r \text{ rounds, } \mathrm{rerr}(\mathcal{P}) \leq \varepsilon, \mathrm{verr}(\mathcal{P}) \leq \delta\}\,,\ \text{where}\\
\mathrm{cost}^{*}(\mathcal{P}) &:= \max\nolimits_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \mathrm{cost}^{*}(\mathcal{P}; x, y)\,.
\end{aligned}$$

Notice that refutation, verification, and overall complexities use (expected) communication cost as the underlying measure, whereas total complexity uses the (more standard) worst-case communication cost.

▶ **Definition 2** (Information Cost and Complexity)**.** Let $\mathcal{P}$, f, and λ be as above, and suppose the players in $\mathcal{P}$ are allowed to use private coins in addition to a public random string $\mathfrak{R}$. The λ-*information cost* of $\mathcal{P}$ and the r-round λ-*information complexity* of f are defined as follows, respectively:

$$\operatorname{icost}^{\lambda}(\mathcal{P}) := \mathrm{I}(XY : \mathcal{P}(X,Y) \mid \mathfrak{R}),$$
$$\mathrm{IC}^{\lambda,(r)}_{\varepsilon,\delta}(f) := \inf\{\operatorname{icost}^{\lambda}(\mathcal{P}) : \mathcal{P} \text{ uses } r \text{ rounds, } \operatorname{rerr}(\mathcal{P}) \le \varepsilon, \operatorname{verr}(\mathcal{P}) \le \delta\}.$$

where $\mathrm{I}(\_ : \_ \mid \_)$ denotes conditional mutual information. For readers familiar with recent literature on information complexity [5, 6], we note that this is technically the "external" information cost rather than the "internal" one. However, we shall study information costs mostly with respect to a uniform input distribution, and in this setting there is no difference between external and internal information cost.

It has long been known that information complexity lower bounds standard worst-case communication complexity: this was the main reason for defining the notion [15]. The simple proof boils down to

$$\mathrm{I}(XY : \mathcal{P}(X,Y) \mid \mathfrak{R}) \le \mathrm{H}(\mathcal{P}(X,Y)) \le \max|\mathcal{P}(X,Y)|.$$

In our setting, with communication cost defined in the expected sense, it is still the case that

$$\mathrm{IC}^{\lambda,(r)}_{\varepsilon,\delta}(f) \le \mathrm{R}^{(r)}_{\varepsilon,\delta}(f) \tag{1}$$

This time the proof boils down to the inequality $\mathrm{H}(\mathcal{P}(X,Y)) \le \mathbb{E}[\,|\mathcal{P}(X,Y)|\,]$, which follows from Shannon's source coding theorem (see Fact 29 in appendix).

2.1 Summary of Results: Equality

The functions $\textsc{eq}_n$ and $\textsc{oreq}_{n,k}$ have been defined in Section 1 already. To formalize our bounds for these problems, we introduce the iterated logarithm functions $\operatorname{ilog}^k : \mathbb{R}_+ \to \mathbb{R}_+$, which are defined as follows.

$$\operatorname{ilog}^0 z := \max\{1, z\}, \quad \forall\, z \in \mathbb{R}_+,$$
$$\operatorname{ilog}^k z := \max\{1, \log(\operatorname{ilog}^{k-1} z)\}, \quad \forall\, k \in \mathbb{N}, z \in \mathbb{R}_+.$$

For all practical purposes, we may pretend that $\operatorname{ilog}^0 = \mathrm{id}$, and $\operatorname{ilog}^k = \log \circ \operatorname{ilog}^{k-1}$, for $k \in \mathbb{N}$.

We use ξ to denote the uniform distribution on $\{0,1\}^n$, and put $\mu := \xi \otimes \xi$. Thus μ is the uniform distribution on inputs to $\textsc{eq}_n$. Strictly speaking these should be denoted ξ_n and μ_n, but we choose to let n be understood from the context. In all our complexity bounds, we tacitly assume that n is sufficiently large. The various parts of the summary theorems below are proved later in the paper, and we indicate on the right where these detailed proofs can be found.

▶ **Theorem 3** (Zero-Error Bounds)**.** *The complexity of* EQUALITY *satisfies the following bounds:*

1. $\mathrm{R}^{(r),\mathrm{ref}}_{0,0}(\textsc{eq}_n) \le \operatorname{ilog}^{r-1} n + 3$.
2. $\mathrm{R}^{(r),\mathrm{ver}}_{0,0}(\textsc{eq}_n) \le n$.
3. $\mathrm{R}^{(r),\mathrm{ref}}_{0,0}(\textsc{eq}_n) = \mathrm{D}^{\mu,(r),\mathrm{ref}}_{0,0}(\textsc{eq}_n) \ge \operatorname{ilog}^{r-1} n - 1$. [Theorem 32]
4. $\mathrm{R}^{(r),\mathrm{ver}}_{0,0}(\textsc{eq}_n) = \mathrm{D}^{\mu,(r),\mathrm{ver}}_{0,0}(\textsc{eq}_n) \ge n$. [Theorem 35]

Notice that these bounds are almost completely tight, differing at most by the tiny additive constant 4. Next, we allow our protocols some error. We continue to have bounds tight up to an additive constant for the verification cost (the case of one-sided error is especially interesting: just set $\delta = 0$ in the results below), and we have bounds tight up to a multiplicative constant in the other cases. To better appreciate the next several bounds, let us first consider the "trivial" one-round protocol for EQ_n that achieves ε refutation error. This protocol communicates $\min\{n, \log(1/\varepsilon)\}$ bits: it's as though the instance size drops from n to $\min\{n, \log(1/\varepsilon)\}$ when we allow this refutation error. This motivates the following definition.

▶ **Definition 4** (Effective Instance Size). When considering protocols for EQ_n with refutation and verification errors bounded by ε and δ, respectively, we define the effective instance size to be

$$\hat{n} := \min\{n + \log(1-\delta), \log((1-\delta)^2/\varepsilon)\}\,.$$

▶ **Theorem 5** (Two-Sided-Error Bounds). EQUALITY *satisfies the following:*

5. $\mathrm{R}^{(r),\mathrm{ref}}_{\varepsilon,\delta}(\text{EQ}_n) \le (1-\delta)\,\mathrm{ilog}^{r-1}\,\hat{n} + 5.$ [Corollary 26]
6. $\mathrm{R}^{(r),\mathrm{ver}}_{\varepsilon,\delta}(\text{EQ}_n) \le (1-\delta)\hat{n} + 3.$ [Corollary 27]
7. $\mathrm{D}^{\mu,(r),\mathrm{ver}}_{\varepsilon,\delta}(\text{EQ}_n) \ge (1-\delta)(\hat{n}-1)\ .$ [Theorem 43]
8. $\mathrm{R}^{(r),\mathrm{ver}}_{\varepsilon,\delta}(\text{EQ}_n) \ge \frac{1}{8}(1-\delta)^2(\hat{n} + \log(1-\delta) - 5).$ [Theorem 44]
9. $\mathrm{D}^{\mu,(r),\mathrm{ref}}_{\varepsilon,\delta}(\text{EQ}_n) = \Omega((1-\delta)^2\,\mathrm{ilog}^{r-1}\,\hat{n})$. *This bound holds for all* ε, δ *such that* $\delta \le 1 - 2^{-n/2}$ *and* $\varepsilon/(1-\delta)^2 < 1/8$. [Theorem 41]
10. $\mathrm{R}^{(r),\mathrm{ref}}_{\varepsilon,\delta}(\text{EQ}_n) = \Omega((1-\delta)^3\,\mathrm{ilog}^{r-1}\,\hat{n})$. *This bound holds for all* ε, δ *such that* $\delta \le 1 - 2^{-n/2}$ *and* $\varepsilon/(1-\delta)^3 \le 1/64$. [Theorem 42]

Observe that the "constant refutation error" setting $\varepsilon = O(1)$ is not very interesting, as it makes these complexities constant. But observe also that the situation is very different for the verification error, δ: we continue to obtain strong lower bounds even when δ is very close to 1. This is in accordance with our intuition that verification (of equality) is much harder than refutation.

Finally, we turn to information complexity and arrive at the most important result of this paper.

▶ **Theorem 6** (Main Theorem: Information Complexity Bound). *Suppose* $\delta \le 1 - 8(\mathrm{ilog}^{r-2}\,\hat{n})^{-1/8}$. *Then:*

11. $\mathrm{IC}^{\mu,(r)}_{\varepsilon,\delta}(\text{EQ}_n) = \Omega((1-\delta)^3\,\mathrm{ilog}^{r-1}\,\hat{n})$. [Theorem 51]

Applications. As applications, we can recover weaker lower bounds for OR-EQUALITY, DISJOINTNESS, and PRIVATE-INTERSECTION. We emphasize that our main result is our thorough study of EQUALITY, including the direct development of information cost bounds for bounded-round protocols and the analysis of verification vs. refutation error.

2.2 On Yao's Minimax Lemma

Distributional lower bounds imply worst-case randomized ones by an averaging argument that constitutes the "easy" direction of Yao's minimax lemma [43]. Yet, in Theorem 5 we claim somewhat weaker randomized bounds than the corresponding distributional ones. The reason is that in our setting, the averaging argument will need to fix the random coins of a protocol so as to preserve multiple measures (e. g., refutation error as well as cost).

Though this is easily accomplished, we pay a penalty of small constant factor increase in our measures.

Ironically, the "hard" direction of Yao's minimax lemma is particularly easy in the case of EQ_n, because EQUALITY is in a sense *uniform self-reducible*. See Theorem 25, where we show how to turn a protocol designed for the uniform distribution into a randomized one with worst-case guarantees. In this way, *the uniform distribution is provably the hardest distribution for* EQUALITY.

3 Main Theorem: Bounded-Round Information Complexity of Equality

In this section we prove Theorem 6, which we think of as the most important result of this paper. We wish to lower bound the bounded-round information complexity of EQUALITY with respect to the uniform distribution. Recall that we are concerned chiefly with protocols that achieve very low refutation error, though they may have rather high verification error. We will prove our lower bound by proving a round elimination lemma for EQ_n that targets *information* cost, and then applying this lemma repeatedly.

This proof has much more technical complexity than our other lower bound proofs. Let us see why. There are two main technical difficulties and they arise, ultimately, from the same source: the inability to use (the easy direction of) Yao's minimax lemma. When proving a lower bound on *communication* cost, Yao's lemma allows us to fix the random string used by any purported protocol, which immediately moves us into the clean world of deterministic protocols. This hammer is unavailable to us when working with *information* cost. The most we can do is to "average away" the public randomness. We then have to deal with (private coin) randomized protocols the entire way through the round elimination argument. As a result, our intermediate protocols, obtained by eliminating some rounds of our original protocol, do not obey straightforward cost and error guarantees. This is the first technical difficulty, and our solution to it leads us to the concept of a "kernel" in Definition 7 below.

The second technical difficulty is that we are unable to switch to the simpler case of zero verification error like we did in the proof of Theorem 5, Parts (9) and (10). Therefore, all our intermediate protocols continue to have verification error. Since errors scale up with each round elimination, and the verification error starts out high, we cannot afford even a constant-factor scaling. We must play very delicately with our error parameters, which leads us to the somewhat complicated parametrization seen in Definition 8 below.

3.1 The Round Elimination Argument

A standard round elimination argument works by showing that if there is a "good" r-round protocol, then there exists a "good" $(r-1)$-round protocol. What it means to be a "good" protocol is typically parameterized, with the parameters degrading each time a round is eliminated. The trick is to carefully control how the parameters degrade, so that after all communication has been eliminated, a nontrival problem instance remains.

We follow the same approach for our round elimination argument. Central to our parameterization of EQUALITY protocols is the notion of a *kernel*. Roughly speaking, we start by assuming there is an r-round protocol for inputs that are nearly uniformly distributed over some set S, and we show that after elminating the first message, we can construct a protocol for inputs nearly uniformly distributed over a set $S' \subseteq S$. The sets S, S' are our *kernels*, and they capture where the remaining "action" is.

▶ **Definition 7** (Kernel). Let p and q be probability distributions on $\{0,1\}^n$, let $S \subseteq \{0,1\}^n$, and let $\ell \geq 0$ be a real number. The triple (p, q, S) is defined to be an *ℓ-kernel* if the following properties hold.

[K1] $\mathrm{H}(p) \geq n - \ell$ and $\mathrm{H}(q) \geq n - \ell$.
[K2] $p(S) \geq 2^{-\ell}$ and $q(S) \geq \frac{1}{2}$.
[K3] For all $x \in S$ we have $q(x) \geq 2^{-n-\ell}$.

▶ **Definition 8** (Parametrized Protocols). Suppose we have an integer $r \geq 1$, and nonnegative reals ℓ, a, b, and c. A protocol $\mathcal{P}$ for EQ_n is defined to be an $[r, \ell, a, b, c]$-protocol if there exists an ℓ-kernel (p, q, S) such that the following properties hold.

[P1] The protocol $\mathcal{P}$ is private-coin and uses r rounds, with Alice speaking in the first round.
[P2] We have $\mathrm{err}^{p\otimes q|S\times S}(\mathcal{P}) = \Pr_{(X,Y)\sim p\otimes q}[\mathrm{out}(\mathcal{P}(X,Y)) \neq \mathrm{EQ}_n(X,Y) \mid (X,Y) \in S \times S] \leq 2^{-a}$.
[P3] We have $\mathrm{verr}^{p\otimes\xi|S\times S}(\mathcal{P}) = \Pr_{X\sim p}[\mathrm{out}(\mathcal{P}(X,X)) = 0 \mid X \in S] \leq 1 - 2^{-b}$.
[P4] We have $\mathrm{icost}^{p\otimes q}(\mathcal{P}) \leq c$.

We alert the reader to the fact that [P2] considers overall error, and not refutation error. We encourage the reader to take a careful look at [P3] and verify the equality claimed therein. It is straightforward, once one revisits Definition 1 and recalls that ξ denotes the uniform distribution on $\{0,1\}^n$.

Since we have a number of parameters at play, it is worth recording the following simple observation.

▶ **Fact 9.** *Suppose that $\ell' \geq \ell, c' \geq c, a' \leq a$, and $b' \geq b$. Then every ℓ-kernel is also an ℓ'-kernel, and every $[r, \ell, a, b, c]$-protocol is also an $[r, \ell', a', b', c']$-protocol.* ◀

▶ **Theorem 10** (Information-Theoretic Round Elimination for EQUALITY). *If there exists an $[r, \ell, a, b, c]$-protocol with $r \geq 1$ and $c \geq 4$, then there exists an $[r-1, \ell', a', b', c']$-protocol, where*

$$\ell' := (c+\ell)2^{\ell+2b+7}, \qquad a' := a - (c+\ell)2^{\ell+2b+8},$$
$$b' := b + 2, \qquad c' := (c+2)2^{\ell+2b+6}.$$

Proof. Let $\mathcal{P}$ be an $[r, \ell, a, b, c]$-protocol, and let (p, q, S) be an ℓ-kernel satisfying the conditions in Definition 8. Assume WLOG that each message in $\mathcal{P}$ is generated using a fresh random string. Let $X \sim p$ and $Y \sim q$ be independent random variables denoting an input to $\mathcal{P}$. Let $M_1, \ldots, M_r$ be random variables denoting the messages sent in $\mathcal{P}$ on input (X, Y), with M_j being the jth message; note that these variables depend on X, Y, and the random strings used by the players. We then have

$$c \geq \mathrm{icost}^{p\otimes q}(\mathcal{P}) = \mathrm{I}(XY : M_1 M_2 \ldots M_r) = \mathrm{I}(X : M_1) + \mathrm{I}(XY : M_2 \ldots M_r \mid M_1), \tag{2}$$

where the final step uses the chain rule for mutual information, and the fact that M_1 and Y are independent. In particular, we have $\mathrm{I}(X : M_1) \leq c$, and so $\mathrm{H}(X \mid M_1) = \mathrm{H}(X) - \mathrm{I}(X : M_1) \geq n - \ell - c$. By Lemma 19,

$$\mathrm{H}(X \mid M_1, X \in S) \geq n - \frac{\ell + c + 1}{p(S)} \geq n - (\ell + c + 1)2^{\ell}. \tag{3}$$

Let $\mathcal{M}$ be the set of messages that Alice sends with positive probability as her first message in $\mathcal{P}$, given the random input X, i. e., $\mathcal{M} := \{\mathfrak{m} : \Pr[M_1 = \mathfrak{m}] > 0\}$. Consider

a particular message $\mathfrak{m} \in \mathcal{M}$. Let $\mathcal{P}'_{\mathfrak{m}}$ denote the following protocol for EQ_n. The players simulate $\mathcal{P}$ on their input, except that Alice is assumed to have sent $\mathfrak{m}$ as her first message. As a result, $\mathcal{P}'_{\mathfrak{m}}$ has $r-1$ rounds and Bob is the player to send the first message in $\mathcal{P}'_{\mathfrak{m}}$. Let $\pi_{\mathfrak{m}}$ and q' be the distributions of $(X \mid M_1 = \mathfrak{m} \wedge X \in S)$ and $(Y \mid Y \in S)$, respectively.

Observe that $\text{icost}^{\pi_{\mathfrak{m}} \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) = \text{I}(XY : M_2 \dots M_r \mid M_1 = \mathfrak{m} \wedge (X,Y) \in S \times S)$. Letting L denote a random first message distributed identically to M_1, we now get

$$\begin{aligned}\mathbb{E}_L\left[\text{icost}^{\pi_L \otimes q'}(\mathcal{P}'_L)\right] &= \text{I}(XY : M_2 \dots M_r \mid M_1, (X,Y) \in S \times S) \\ &\leq \frac{\text{I}(XY : M_2 \dots M_r \mid M_1) + 1}{p(S)q(S)} \leq (c+1)2^{\ell+1}\,, \end{aligned} \tag{4}$$

where the first inequality uses Lemma 18 and the second inequality uses (2) and Property [K2]. Examining Properties [P2] and [P3], we obtain

$$\mathbb{E}_L\left[\text{err}^{\pi_L \otimes q'}(\mathcal{P}'_L)\right] = \text{err}^{p \otimes q | S \times S}(\mathcal{P}) \leq 2^{-a}\,, \tag{5}$$

$$\mathbb{E}_L\left[\text{verr}^{\pi_L \otimes \xi}(\mathcal{P}'_L)\right] = \text{verr}^{p \otimes \xi | S \times S}(\mathcal{P}) \leq 1 - 2^{-b}\,. \tag{6}$$

▶ **Definition 11** (Good Message)**.** A message $\mathfrak{m} \in \mathcal{M}$ is said to be *good* if the following properties hold:

[G1] $\text{H}(\pi_{\mathfrak{m}}) = \text{H}(X \mid M_1 = \mathfrak{m} \wedge X \in S) \geq n - (\ell + c + 1)2^{\ell+b+3}$,
[G2] $\text{icost}^{\pi_{\mathfrak{m}} \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) \leq 2^{\ell+b+4}(c+1)$,
[G3] $\text{err}^{\pi_{\mathfrak{m}} \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) \leq 2^{-a+b+3}$,
[G4] $\text{verr}^{\pi_{\mathfrak{m}} \otimes \xi}(\mathcal{P}'_{\mathfrak{m}}) \leq 1 - 2^{-b-1}$.

Notice that for all $\mathfrak{m} \in \mathcal{M}$ we have $\text{H}(X \mid M_1 = \mathfrak{m}, X \in S) \leq n$. Hence, viewing (3), (4), (5) and (6) as upper bounds on the expected values of certain nonnegative functions of L, we may apply Markov's inequality to these four conditions and conclude that

$$\Pr[L \text{ is good}] \geq 1 - 2^{-b-3} - 2^{-b-3} - 2^{-b-3} - \frac{1 - 2^{-b}}{1 - 2^{-b-1}} \geq 2^{-b-1} - 3 \cdot 2^{-b-3} > 0\,.$$

Thus, there exists a good message. *From now on, we fix $\mathfrak{m}$ to be such a good message.*

We may rewrite the left-hand side of [G4] as $\mathbb{E}_{Z \sim \pi_{\mathfrak{m}}}[\Pr[\text{out}(\mathcal{P}'_{\mathfrak{m}}(Z,Z)) = 0]]$. So if we define the set $T := \left\{x \in S : \Pr[\text{out}(\mathcal{P}'_{\mathfrak{m}}(x,x)) = 0] \leq 1 - 2^{-b-2}\right\}$ and apply Markov's inequality again, we obtain

$$\pi_{\mathfrak{m}}(T) \geq 1 - \frac{1 - 2^{-b-1}}{1 - 2^{-b-2}} \geq 2^{-b-2}\,. \tag{7}$$

Defining the distribution $p' := \pi_{\mathfrak{m}} \mid T$ and the set $S' := \{x \in T : p'(x) \geq 2^{-n-\ell'}\}$, we now make two claims.

Claim 1: The triple (q', p', S') is an ℓ'-kernel.
Claim 2: We have $\text{err}^{p' \otimes q' | S' \times S'}(\mathcal{P}'_{\mathfrak{m}}) \leq 2^{-a'}$, $\text{verr}^{q' \otimes \xi | S' \times S'}(\mathcal{P}'_{\mathfrak{m}}) \leq 1 - 2^{-b'}$, and $\text{icost}^{p' \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) \leq c'$.

We prove these claims in Appendix D. Notice that these claims essentially say that $\mathcal{P}'_{\mathfrak{m}}$ has all the properties listed in Definition 8, except that Bob starts $\mathcal{P}'_{\mathfrak{m}}$. Interchanging the roles of Alice and Bob in $\mathcal{P}'_{\mathfrak{m}}$ gives us the desired $[r-1, \ell', a', b', c']$-protocol, which completes the proof of the theorem. ◀

The following easy corollary of Theorem 10 will be useful shortly; we defer its proof to Appendix D.

▶ **Corollary 12.** *Let $\tilde{n}, j, r \in \mathbb{N}$ and $a, b \in \mathbb{R}$ with $\tilde{n}$ sufficiently large, $j \geq 1$, $r \geq 1$, and $b \geq 0$. Suppose there exists an $[r, \ell, a-\ell, b, \ell]$-protocol, with $b \leq \ell = \frac{1}{8}\operatorname{ilog}^j \tilde{n}$. Then there exists an $[r-1, \ell', a-\ell', b+2, \ell']$-protocol with $b+2 \leq \ell' = (\operatorname{ilog}^{j-1}\tilde{n})^{1/2} \leq \frac{1}{8}\operatorname{ilog}^{j-1}\tilde{n}$.*

3.2 Finishing the Proof

We are now ready to state and prove the main lower bound on protocols with two-sided error.

▶ **Theorem 13** (Restatement of Main Theorem). *Let $\tilde{n} = \min\{n + \log(1-\delta), \log((1-\delta)/\varepsilon)\}$. Suppose $\delta \leq 1 - 8(\operatorname{ilog}^{r-2}\tilde{n})^{-1/8}$. Then we have $\mathrm{IC}^{\mu,(r)}_{\varepsilon,\delta}(\mathrm{EQ}_n) = \Omega((1-\delta)^3 \operatorname{ilog}^{r-1}\tilde{n})$.*

Proof. We may assume that $r \leq \log^* \tilde{n}$, for otherwise there is nothing to prove. The slight difference between $\tilde{n}$ above and $\hat{n}$, as in Definition 4, is insignificant and can be absorbed by the $\Omega(\cdot)$ notation.

Suppose, to the contrary, that there exists an r-round randomized protocol $\mathcal{P}^*$ for EQ_n, with $\mathrm{rerr}^\mu(\mathcal{P}^*) \leq \varepsilon$, $\mathrm{verr}^\mu(\mathcal{P}^*) \leq \delta$ and $\mathrm{icost}^\mu(\mathcal{P}^*) \leq 2^{-16}(1-\delta)^3 \operatorname{ilog}^{r-1}\tilde{n}$. Recall that we denote the uniform distribution on $\{0,1\}^n$ by ξ and that $\mu = \xi \otimes \xi$. We have

$$\mathrm{err}^\mu(\mathcal{P}^*) = (1-2^{-n})\,\mathrm{rerr}^\mu(\mathcal{P}^*) + 2^{-n}\,\mathrm{verr}^\mu(\mathcal{P}^*) \leq \varepsilon + 2^{-n}(\delta - \varepsilon) \leq \varepsilon + 2^{-n}\,.$$

Let $\mathcal{P}^*_s$ be the private-coin protocol for EQ_n obtained from $\mathcal{P}^*$ by fixing the public random string of $\mathcal{P}^*$ to be s. We have $\mathbb{E}_s[\mathrm{err}^\mu(\mathcal{P}^*_s)] \leq \varepsilon + 2^{-n}$, $\mathbb{E}_s[\mathrm{verr}^\mu(\mathcal{P}^*_s)] \leq \delta$, and $\mathbb{E}_s[\mathrm{icost}(\mathcal{P}^*_s)] \leq 2^{-16}(1-\delta)^3 \operatorname{ilog}^{r-1}\tilde{n}$. By Markov's inequality, there exists s such that $\mathcal{P}^*_s$ simultaneously has $\mathrm{err}^\mu(\mathcal{P}^*_s) \leq 4(\varepsilon + 2^{-n})/(1-\delta)$, $\mathrm{verr}^\mu(\mathcal{P}^*_s) \leq (1+\delta)/2$, and $\mathrm{icost}(\mathcal{P}^*_s) \leq 2^{-14}(1-\delta)^2 \operatorname{ilog}^{r-1}\tilde{n}$: this is because

$$1 - \frac{1-\delta}{4} - \frac{2\delta}{1+\delta} - \frac{1-\delta}{4} = \frac{(1-\delta)^2}{2(1+\delta)} > 0\,.$$

Let $\mathcal{P} = \mathcal{P}^*_s$ for this s. Then $(\xi, \xi, \{0,1\}^n)$ is a 0-kernel and $\mathcal{P}$ is an $[r,\ 0,\ \log\frac{1-\delta}{4(\varepsilon+2^{-n})},\ \log\frac{2}{1-\delta},\ 2^{-14}(1-\delta)^2 \operatorname{ilog}^{r-1}\tilde{n}]$-protocol. Recalling Fact 9 and using $\log\frac{1-\delta}{\varepsilon+2^{-n}} \geq \tilde{n} - 1$, we see that

$$\mathcal{P} \text{ is an } \left[r,\ 0,\ \tilde{n}-3,\ \log\tfrac{1}{1-\delta}+1,\ 2^{-14}(1-\delta)^2 \operatorname{ilog}^{r-1}\tilde{n}\right]\text{-protocol.}$$

Put $\ell_j := \frac{1}{8}\operatorname{ilog}^j \tilde{n}$ for $j \in \mathbb{N}$. Applying round elimination (Theorem 10) to $\mathcal{P}$ and weakening the resulting parameters (using Fact 9) gives us an $[r-1, \ell_{r-1}, \tilde{n} - \ell_{r-1}, \log\frac{1}{1-\delta} + 3, \ell_{r-1}]$-protocol $\mathcal{P}'$.

The upper bound on δ gives us $\log\frac{1}{1-\delta} + 3 \leq \ell_{r-1}$, and so the conditions for Corollary 12 apply. Starting with $\mathcal{P}'$ and applying that corollary repeatedly, each time using the looser estimate on ℓ' in that corollary, we obtain a sequence of protocols with successively fewer rounds. Eventually we reach a $[1, \ell_1, \tilde{n} - \ell_1, \log\frac{1}{1-\delta} + 2(r-1) + 1, \ell_1]$-protocol. Applying Theorem 10 one more time, and using the tighter estimate on ℓ' this time, we get a $[0,\ \tilde{n}^{1/2},\ \tilde{n} - \tilde{n}^{1/2},\ \log\frac{1}{1-\delta} + 2r + 1,\ \tilde{n}^{1/2}]$-protocol $\mathcal{Q}$. Weakening parameters again, we see that $\mathcal{Q}$ is a $[0,\ \tilde{n}^{1/2},\ \frac{1}{2}\tilde{n},\ \frac{1}{3}\log\tilde{n},\ \tilde{n}^{1/2}]$-protocol. Let (p, q, S) be the $\tilde{n}^{1/2}$-kernel for $\mathcal{Q}$. By Property [K1], we have $\mathrm{H}(q) \geq n - \tilde{n}^{1/2}$. Using Lemma 19 and Property [K2], we then have

$$\mathrm{H}(q \mid S) \geq n - \frac{\tilde{n}^{1/2}+1}{q(S)} \geq n - (2\tilde{n}^{1/2} + 2)\,. \tag{8}$$

Since $\mathcal{Q}$ involves no communication, it must behave identically on any two input distributions that have the same marginal on Alice's input. In particular, this gives us the following

crucial equation:

$$\Pr_{X\sim p}[\mathrm{out}(\mathcal{Q}(X,X))=1 \mid X\in S] = \Pr_{(X,Y)\sim p\otimes q}[\mathrm{out}(\mathcal{Q}(X,Y))=1 \mid (X,Y)\in S\times S]\,. \tag{9}$$

Let α denote the above probability. Considering the left-hand side of (9), we have

$$\alpha = 1-\mathrm{verr}^{p\otimes\xi|S\times S}(\mathcal{Q}) \geq 2^{-\frac{1}{3}\log\tilde{n}} = \tilde{n}^{-1/3}\,. \tag{10}$$

On the other hand, whenever $\mathcal{Q}$ outputs 1 on an input (x,y), then either $x=y$ or $\mathcal{Q}$ errs on (x,y). Therefore, considering the right-hand side of (9), we have

$$\begin{aligned}
\alpha &\leq \Pr_{(X,Y)\sim p\otimes q}[X=Y \mid (X,Y)\in S\times S] + \\
&\qquad \Pr_{(X,Y)\sim p\otimes q}[\mathrm{out}(\mathcal{P}(X,Y)) \neq \mathrm{EQ}_n(X,Y) \mid (X,Y)\in S\times S] \\
&\leq \max_{x\in S} \Pr_{Y\sim q|S}[Y=x] + \mathrm{err}^{p\otimes q|S\times S}(\mathcal{Q}) \\
&\leq \frac{2\tilde{n}^{1/2}+3}{n} + 2^{-\frac{1}{2}\tilde{n}} \qquad (11)\\
&\leq 2\tilde{n}^{-1/2} + 3\tilde{n}^{-1} + 2^{-\frac{1}{2}\tilde{n}}\,, \qquad (12)
\end{aligned}$$

where (11) follows from (8) by applying Lemma 20, and (12) uses $\tilde{n}\leq n$.

The bounds (10) and (12) are in contradiction for sufficiently large $\tilde{n}$, which completes the proof. ◀

References

1 Farid Ablayev. Lower bounds for one-way probabilistic communication complexity and their application to space complexity. *Theoretical Computer Science*, 175(2):139–159, 1996.

2 Anil Ada, Arkadev Chattopadhyay, Stephen A. Cook, Lila Fontes, Michal Koucký, and Toniann Pitassi. The hardness of being private. In *IEEE Conference on Computational Complexity*, pages 192–202, 2012.

3 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999. Preliminary version in *Proc. 28th Annual ACM Symposium on the Theory of Computing*, pages 20–29, 1996.

4 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.

5 Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. In *Proc. 41st Annual ACM Symposium on the Theory of Computing*, pages 67–76, 2010.

6 Mark Braverman. Interactive information complexity. In *Proc. 44th Annual ACM Symposium on the Theory of Computing*, pages 505–524, 2012.

7 Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication. In *Proc. 45th Annual ACM Symposium on the Theory of Computing*, 2013. to appear.

8 Mark Braverman and Ankur Moitra. An information complexity approach to extended formulations. In *Proc. 45th Annual ACM Symposium on the Theory of Computing*, 2013. to appear.

9 Mark Braverman and Anup Rao. Information equals amortized communication. In *Proc. 52nd Annual IEEE Symposium on Foundations of Computer Science*, pages 748–757, 2011.

10 Joshua Brody, Amit Chakrabarti, and Ranganath Kondapally. Certifying equality with limited interaction. Technical Report TR12-153, ECCC, 2012.

11 Harry Buhrman, David García-Soriano, Arie Matsliah, and Ronald de Wolf. The non-adaptive query complexity of testing k-parities. *arXiv preprint arXiv:1209.3849*, 2012.

12 Amit Chakrabarti, Graham Cormode, Ranganath Kondapally, and Andrew McGregor. Information cost tradeoffs for augmented index and streaming language recognition. In *Proc. 51st Annual IEEE Symposium on Foundations of Computer Science*, pages 387–396, 2010.

13 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *Proc. 18th Annual IEEE Conference on Computational Complexity*, pages 107–117, 2003.

14 Amit Chakrabarti and Ranganath Kondapally. Everywhere-tight information cost tradeoffs for augmented index. In *Proc. 15th International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 448–459, 2011.

15 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew C. Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *Proc. 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 270–278, 2001.

16 Amit Chakrabarti and Anna Shubina. Nearly private information retrieval. In *Proc. 32nd International Symposium on Mathematical Foundations of Computer Science*, volume 4708 of *Lecture Notes in Computer Science*, pages 383–393, 2007.

17 Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, second edition, 2006.

18 Anirban Dasgupta, Ravi Kumar, and D. Sivakumar. Sparse and lopsided set disjointness via information theory. In *16th International workshop on Randomization*, volume 7409, pages 517–528, 2012.

19 Ronald Fagin, Moni Naor, and Peter Winkler. Comparing information without leaking it. *Commun. ACM*, 39(5):77–85, 1996.

20 Tomas Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized communication complexity. *SIAM J. Comput.*, 24(4):736–750, 1995. Preliminary version in *Proc. 32nd Annual IEEE Symposium on Foundations of Computer Science*, pages 239–248, 1991.

21 Michael J. Freedman, Kobbi Nissim, and Benny Pinkas. Efficient private matching and set intersection. In *EUROCRYPT*, pages 1–19, 2004.

22 Rusins Freivalds. Probabilistic machines can use less running time. In *IFIP Congress*, pages 839–842, 1977.

23 Andre Gronemeier. Asymptotically optimal lower bounds on the NIH-multi-party information complexity of the AND-function and disjointness. In *Proc. 26th International Symposium on Theoretical Aspects of Computer Science*, pages 505–516, 2009.

24 Prahladh Harsha, Rahul Jain, David McAllester, and Jaikumar Radhakrishnan. The communication complexity of correlation. In *Proc. 22nd Annual IEEE Conference on Computational Complexity*, pages 10–23, 2007.

25 Johan Håstad and Avi Wigderson. The randomized communication complexity of set disjointness. *Theory of Computing*, pages 211–219, 2007.

26 Rahul Jain. New strong direct product results in communication complexity. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:24, 2011.

27 Rahul Jain, Attila Pereszlényi, and Penghui Yao. A direct product theorem for the two-party bounded-round public-coin communication complexity. In *Proc. 53rd Annual IEEE Symposium on Foundations of Computer Science*, pages 167–176, 2012.

28 Rahul Jain, Pranab Sen, and Jaikumar Radhakrishnan. Optimal direct sum and privacy trade-off results for quantum and classical communication complexity. *CoRR*, abs/0807.1267, 2008.

29 Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Disc. Math.*, 5(4):547–557, 1992.

30 Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes. *J. Comput. Syst. Sci.*, 69(3):395–420, 2004. Preliminary version in *Proc. 35th Annual ACM Symposium on the Theory of Computing*, pages 106–115, 2003.

31 Hartmut Klauck. On quantum and approximate privacy. In *STACS*, pages 335–346, 2002.

32 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, Cambridge, 1997.

33 Eyal Kushilevitz and Enav Weinreb. The communication complexity of set-disjointness with small sets and 0-1 intersection. In *Proc. 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 63–72, 2009.

34 Frédéric Magniez, Claire Mathieu, and Ashwin Nayak. Recognizing well-parenthesized expressions in the streaming model. In *Proc. 41st Annual ACM Symposium on the Theory of Computing*, pages 261–270, 2010.

35 Kurt Mehlhorn and Erik M. Schmidt. Las Vegas is better than determinism in VLSI and distributed computing (extended abstract). In *Proc. 14th Annual ACM Symposium on the Theory of Computing*, pages 330–337, 1982.

36 Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. *J. Comput. Syst. Sci.*, 57(1):37–49, 1998. Preliminary version in *Proc. 27th Annual ACM Symposium on the Theory of Computing*, pages 103–111, 1995.

37 M. Molinaro, D.P. Woodruff, and G. Yaroslavtsev. Beating the direct sum theorem in communication complexity with implications for sketching. In *Proc. 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2013.

38 Moni Naor and Benny Pinkas. Oblivious polynomial evaluation. *SIAM J. Comput.*, 35(5):1254–1281, 2006.

39 Ashwin Nayak. Optimal lower bounds for quantum automata and random access codes. In *Proc. 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 124–133, 1999.

40 Mihai Pătraşcu. Unifying the landscape of cell-probe lower bounds. *SIAM J. Comput.*, 40(3):827–847, 2011.

41 Alexander Razborov. On the distributional complexity of disjointness. *Theor. Comput. Sci.*, 106(2):385–390, 1992. Preliminary version in *Proc. 17th International Colloquium on Automata, Languages and Programming*, pages 249–253, 1990.

42 Mert Saglam and Gábor Tardos. On the communication complexity of sparse set disjointness and exists-equal problems. In *Proc. 54th Annual IEEE Symposium on Foundations of Computer Science*, pages 678–687, 2013.

43 Andrew C. Yao. Probabilistic computations: Towards a unified measure of complexity. In *Proc. 18th Annual IEEE Symposium on Foundations of Computer Science*, pages 222–227, 1977.

44 Andrew C. Yao. Some complexity questions related to distributive computing. In *Proc. 11th Annual ACM Symposium on the Theory of Computing*, pages 209–213, 1979.

A Information theory

A.1 Basic Probability, Properties of Entropy and Mutual Information

We will use the following fact about collision probability of a random function.

▶ **Fact 14.** *Given a subset $S \subseteq [n]$ for size $|S| \geq 2$, $i \geq 0$ and $t = \Theta(|S|^{i+2})$, a random function $h\colon [n] \to [t]$ has no collisions with probability at least $1 - 1/|S|^i$, namely for all $x, y \in S$ such that $x \neq y$ it holds that $h(x) \neq h(y)$. Moreover, a random hash function satisfying such guarantee can be constructed using only $O(\log n)$ random bits.*

▶ **Definition 15.** Let λ be a probability distribution on a finite set S and let $T \subseteq S$ be an event with $\lambda(T) \neq 0$. We write $\lambda \mid T$ to denote the distribution obtained by conditioning λ on T. To be explicit, $\lambda \mid T$ is given by

$$(\lambda \mid T)(x) = \begin{cases} 0\,, & \text{if } x \notin T\,, \\ \lambda(x)/\lambda(T)\,, & \text{if } x \in T\,. \end{cases}$$

Also, we write $\mathrm{H}(\lambda)$ to denote the entropy of a random variable distributed according to λ, i. e., $\mathrm{H}(\lambda) = \mathrm{H}(X)$, where $X \sim \lambda$.

▶ **Lemma 16** (Equivalent to Lemma 30)**.** *With λ, S and T as above, let $f : S \to \mathbb{R}_+$ be a nonnegative function. Then $\mathbb{E}_{X\sim\lambda|T}[f(X)] \leq \mathbb{E}_{X\sim\lambda}[f(X)]/\lambda(T)$.* ◀

We give a summary of basic properties of the entropy of a discrete random variable X, denoted as $H(X)$, and the mutual information between two discrete random variables X and Y, denoted as $\mathrm{I}(X : Y) = H(X) - H(X \mid Y)$, below (see Chapter 2 in [17] for the proofs). We denote the support of a random variable X as $supp(X)$.

▶ **Proposition 17.**
1. *Entropy span:* $0 \leq H(X) \leq \log |supp(X)|$.
2. $\mathrm{I}(X : Y) \geq 0$ *because* $H(X \mid Y) \leq H(X)$.
3. *Chain rule:* $\mathrm{I}(X_1, X_2, \ldots, X_n : Y \mid Z) = \sum_{i=1}^{n} \mathrm{I}(X_i : Y \mid X_1, \ldots, X_{i-1}, Z)$.
4. *Subadditivity:* $H(X, Y \mid Z) \leq H(X \mid Z) + H(Y \mid Z)$, *where the equality holds if and only if X and Y are independent conditioned on Z.*
5. *Fano's inequality: Let A be a random variable, which can be used as "predictor" of X, namely there exists a function g such that $\Pr[g(A) = X] \geq 1 - \delta$ for some $\delta < 1/2$. If $|supp(X)| \geq 2$ then $H(X \mid A) \leq \delta \log(|supp(X)| - 1) + h_2(\delta)$, where $h_2(\delta) = \delta \log(1/\delta) + (1 - \delta) \log \frac{1}{1-\delta}$ is the binary entropy.*

▶ **Lemma 18.** *Let Z, W be jointly distributed random variables. Let $\mathcal{E}$ be an event. Then,*

$$\mathrm{I}(Z : W) \geq \Pr[\mathcal{E}]\ \mathrm{I}(Z : W \mid \mathcal{E}) - 1\,.$$

Proof. Let D be the indicator random variable for $\mathcal{E}$. Then we have

$$\mathrm{I}(Z : W \mid D) = \Pr[\mathcal{E}]\ \mathrm{I}(Z : W \mid \mathcal{E}) + \Pr[\neg\mathcal{E}]\ \mathrm{I}(Z : W \mid \neg\mathcal{E}) \geq \Pr[\mathcal{E}]\ \mathrm{I}(Z : W \mid \mathcal{E})\,. \tag{13}$$

Note that $\mathrm{I}(Z : D \mid W) \leq \mathrm{H}(D \mid W) \leq \mathrm{H}(D) \leq 1$. Using the chain rule for mutual information twice, we get

$$\mathrm{I}(Z : W \mid D) \leq \mathrm{I}(Z : WD) = \mathrm{I}(Z : W) + \mathrm{I}(Z : D \mid W) \leq \mathrm{I}(Z : W) + 1\,. \tag{14}$$

The lemma follows by combining inequalities (13) and (14). ◀

To appreciate the next two lemmas, it will help to imagine that $d \ll n$.

▶ **Lemma 19.** *Let Z, W be jointly distributed random variables, with Z taking values in $\{0,1\}^n$, and let $\mathcal{E}$ be an event. Then*

$$\mathrm{H}(Z \mid W) \geq n - d \implies \mathrm{H}(Z \mid W, \mathcal{E}) \geq n - (d+1)/\Pr[\mathcal{E}]\,.$$

In particular, taking W to be a constant, we have $\mathrm{H}(Z) \geq n - d \implies \mathrm{H}(Z \mid \mathcal{E}) \geq n - (d+1)/\Pr[\mathcal{E}]$.

Proof. We use the fact that the entropy of Z can be at most n, even after arbitrary conditioning. This gives

$$\begin{aligned} n - d &\leq \mathrm{H}(Z \mid W) \\ &= \Pr[\mathcal{E}]\ \mathrm{H}(Z \mid W, \mathcal{E}) + (1 - \Pr[\mathcal{E}])\,\mathrm{H}(Z \mid W, \neg\mathcal{E}) + \mathrm{H}_b(\Pr[\mathcal{E}]) \\ &\leq \Pr[\mathcal{E}]\ \mathrm{H}(Z \mid W, \mathcal{E}) + (1 - \Pr[\mathcal{E}])n + 1\,, \end{aligned}$$

where $\mathrm{H}_b(x) := -x \log x - (1-x)\log(1-x)$. The lemma follows by rearranging the above inequality. ◀

▶ **Lemma 20.** *Let Z be a random variable taking values in $\{0,1\}^n$ and let $z \in \{0,1\}^n$. Then*

$$\mathrm{H}(Z) \geq n - d \implies \Pr[Z = z] \leq (d+1)/n\,.$$

Proof. The lemma follows by rearranging the following inequality, which is a consequence of Lemma 19:

$$0 = \mathrm{H}(Z \mid Z = z) \geq n - \frac{d+1}{\Pr[Z = z]}\,.$$ ◀

A.2 Protocols with Abortion

We recall standard definitions from information complexity and introduce the information complexity for protocols with abortion, denoted as $\mathrm{IC}^{\mu}_{\alpha,\beta,\delta}(f \mid \nu)$. Given a communication problem $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, consider the augmented space $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ for some $\mathcal{D}$. Let λ be a distribution over $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$, which induces marginals μ on $\mathcal{X} \times \mathcal{Y}$ and ν on $\mathcal{D}$. We say that ν *partitions* μ, if μ is a *mixture* of product distributions, namely for a random variable $(X, Y, D) \sim \lambda$, conditioning on any value of D makes the distribution of (X, Y) product.

To simplify the notation, a *δ-protocol* for f is one that for all inputs $(x, y) \in \mathcal{X} \times \mathcal{Y}$ computes $f(x, y)$ with probability at least $1 - \delta$ (over the randomness of the protocol).

▶ **Definition 21** (Protocols with Abortion)**.** Consider a communication problem given by $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ and a probability distribution μ over $\mathcal{X} \times \mathcal{Y}$. We say that a deterministic protocol $\mathcal{P}_D$ *(β, δ)-computes f* with respect to μ if it satisfies the following (where $(X, Y) \sim \mu$):

1. (Abortion probability) $\Pr[\mathcal{P}_D(X, Y) =$ 'abort'$] \leq \beta$
2. (Failure probability) $\Pr[\mathcal{P}_D(X, Y) \neq f(X, Y) \mid \mathcal{P}_D(X, Y) \neq$ 'abort'$] \leq \delta$.

We can view randomized protocols as distributions over deterministic protocols (both for private-coin and public-coin protocols). We say that a randomized protocol $\mathcal{P}$ (α, β, δ)-computes f with respect to μ if $\Pr_{\mathcal{P}_D \sim \mathcal{P}}[\mathcal{P}_D\ (\beta, \delta)\text{-computes } f] \geq 1 - \alpha$. The probability is taken over all randomness of the parties.

Using the definitions above we can now introduce the notion of information complexity for protocols with abortions formally.

▶ **Definition 22** (Information Complexity for Protocols with Abortion). Let $\mathcal{P}$ be a protocol, which computes f. The *conditional information cost* of $\mathcal{P}$ under λ is defined as $\mathrm{I}(\mathcal{P}(X,Y) : X, Y \mid D)$, where $(X,Y,D) \sim \lambda$. The *conditional information complexity* of f with respect to λ, denoted by $\mathrm{IC}_{\mu,\delta}(f \mid \nu)$, is defined as the minimum conditional information cost of a δ-protocol for f. The *information complexity with aborts*, denoted by $\mathrm{IC}^{\mu}_{\alpha,\beta,\delta}(f \mid \nu)$, is the minimum conditional information cost of a protocol that (α,β,δ)-computes f. The analogous quantities $\mathrm{IC}^{\mu,(r)}_{\delta}(f \mid \nu)$ and $\mathrm{IC}^{\mu,(r)}_{\alpha,\beta,\delta}(f \mid \nu)$ are defined by taking the respective minimums over only r-round protocols.

B Upper Bounds

In this section, we provide deterministic and randomized protocols for EQ_n with low refutation cost and low verification cost. Recall Definition 4, which introduced the quantity $\hat{n} = \min\left\{n + \log(1-\delta), \log\frac{(1-\delta)^2}{\varepsilon}\right\}$ as the effective instance size. One can derive one-sided-error and zero-error versions of these results by setting δ and/or ε to zero as needed, and using the convention $\log(w/0) = +\infty$ for $w > 0$. One can in fact tighten the analysis for the case $\varepsilon = \delta = 0$ to obtain the bounds in Theorem 3.

▶ **Theorem 23.** *Suppose $n, r \in \mathbb{N}$ and $\varepsilon, \delta \in [0,1]$ are such that $\delta < 1 - 2^{-n/2}$ and $\operatorname{ilog}^{r-1} \hat{n} \geq 4$. Then*

$$\mathrm{D}^{\mu,(r),\mathrm{ref}}_{\varepsilon,\delta}(\mathrm{EQ}_n) \leq (1-\delta)\operatorname{ilog}^{r-1}\hat{n} + 5\,.$$

Proof. To gain intuition, we first consider $\delta = 0$, in which case we have $\hat{n} = \min\{n, \log(1/\varepsilon)\}$. The basic idea was already outlined in Section 1. Since we need only handle a random input, we do not need fingerprints. Instead, Alice and Bob take turns revealing increasingly longer prefixes of their inputs: in the jth round, the player to speak sends the next $\approx \operatorname{ilog}^{r-j}\hat{n}$ bits of her input. Whenever a player witnesses a mismatch in prefixes, she *aborts* (and the protocol outputs 0). If the protocol ends without an abortion, it outputs 1. The protocol described so far clearly has no false negatives, and after filling in some details (see below), we can show that it has the desired refutation cost and refutation error.

To achieve further savings for nonzero δ, we partition $\{0,1\}^n$ into sets $S, T \subseteq \{0,1\}^n$ such that $|S| \approx (1-\delta)2^n$. Each player aborts the protocol at her first opportunity if her input lies in T. Otherwise, they emulate the above protocol on the smaller input space $S \times S$.

We now make things precise. Set

$$\begin{aligned} n' &:= n + \lceil \log(1-\delta) \rceil\,, \\ n'' &:= \min\{n',\, 2 + \lceil \log((1-\delta)^2/\varepsilon) \rceil\}\,, \\ t_j &:= \begin{cases} \lceil \operatorname{ilog}^{r-j}\hat{n} \rceil\,, & \text{if } 1 \leq j < r\,, \\ n'' - \sum_{j=1}^{r-1} t_j\,, & \text{if } j = r\,. \end{cases} \end{aligned}$$

Choose an arbitrary partition of $\{0,1\}^n$ into subsets S and T such that $|S| = 2^{n'}$. Fix an arbitrary bijection $g : S \to \{0,1\}^{n'}$.

The protocol—which we call $\mathcal{P}$—works as follows on input $(x,y) \in \{0,1\}^n \times \{0,1\}^n$. We write $x[i_1 : i_2]$ to denote the substring $x_{i_1} x_{i_1+1} \ldots x_{i_2}$ of x. Each nonempty message in the protocol will be either the string "0", indicating abortion, or "1" followed by a *payload*

string. Each player maintains a variable ℓ that records the length of the prefix that has been compared so far; initially they set $\ell \leftarrow 0$.

The players keep track of whether an abortion has occurred. Once an abortion occurs, all further messages in the protocol will be empty strings. Once r rounds have been completed, the appropriate player will output 0 if an abortion has occurred, and 1 otherwise.

Round j proceeds as follows. Let $P \in \{\text{Alice, Bob}\}$ be the player who speaks in this round, and let $z \in \{x, y\}$ be their input. If necessary, P aborts if $z \in T$. Now suppose that an abortion has not yet occurred. If $j = 1$, then P sends the substring $g(z)[1 : t_1]$, sets $\ell \leftarrow t_1$, and the round ends. Otherwise, suppose P receives a non-aborting message with payload w. If P finds that $w \neq g(z)[\ell + 1 : \ell + t_{j-1}]$ then she aborts, else if $j < r$, she continues the protocol by sending the next t_j bits of $g(z)$, i. e., she sends $g(z)[\ell + t_{j-1} + 1 : \ell + t_{j-1} + t_j]$, sets $\ell \leftarrow \ell + t_{j-1} + t_j$, and the round ends.

The protocol's logic is shown in pseudocode form below, for readers who prefer that presentation.

Algorithm 1: Round j of the protocol $\mathcal{P}$. Here $t_0 = 0$ and "Round $r + 1$" is the output announcement.

```
if j ≤ r then
    if aborted then send emptystring ;
    else
        if z ∈ T then abort;
        w ← payload of most recently received message ;
        if w ≠ g(z)[ℓ + 1 : ℓ + t_{j−1}] then abort;
        send "1" followed by g(x)[ℓ + t_{j−1} + 1 : ℓ + t_{j−1} + t_j], and set ℓ ← ℓ + t_{j−1} + t_j ;
else
    if aborted then output 0 ;
    else
        w ← payload of most recently received message ;
        if w ≠ g(z)[ℓ + 1 : ℓ + t_{j−1}] then output 0 else output 1 ;
```

It is easy to see that $\operatorname{verr}^\mu(\mathcal{P}) \leq \delta$, since players only abort an (x, x) input when $x \in T$. Next, note that a false positive occurs only when $(x, y) \in S \times S$ and $g(x)[1 : n''] = g(y)[1 : n'']$. When $n'' = n'$ (which corresponds, roughly, to $\varepsilon < (1 - \delta)2^{-n}$), Alice and Bob end up comparing all bits of $g(x)$ and $g(y)$, and we get $\operatorname{rerr}^\mu(\mathcal{P}) = 0$. In the other case, we have $n'' = 2 + \lceil \log((1 - \delta)^2/\varepsilon) \rceil$. Letting $(X, Y) \sim \mu$, we have

$$\begin{aligned}\operatorname{rerr}^\mu(\mathcal{P}) &= \Pr[(X, Y) \in S \times S \mid X \neq Y] \cdot \Pr\left[g(X)[1 : n''] = g(Y)[1 : n''] \mid g(X) \neq g(Y)\right] \\ &\leq \left(2^{n'-n}\right)^2 \cdot \frac{2^{n'-n''} - 1}{2^{n'} - 1} \leq 2^{2\lceil \log(1-\delta) \rceil} \cdot 2^{-n''} \leq 2^{2(1+\log(1-\delta))} \cdot \frac{\varepsilon}{4(1-\delta)^2} = \varepsilon\,.\end{aligned}$$

Finally, we analyze the refutation cost. Let a_j denote the expected total communication in rounds $\geq j$, conditioned on not aborting before round j. For convenience, set $a_{r+1} = 0$. We claim that $a_j \leq 3$ for all $j > 2$ and prove so by induction from $r + 1 \rightsquigarrow 3$. The base case ($j = r + 1$) is trivial. Conditioned on not aborting before the jth round, the player whose turn it is to speak receives t_{j-1} bits to compare with her own input. Estimating as above, this will fail to cause an abortion with probability at most $2^{-t_{j-1}}$. Therefore, the player to speak will send at most 1 bit in this round to indicate abortion (or not) plus, with probability at most $2^{-t_{j-1}}$, will continue the communication, which will cost t_j bits in this

round and a_{j+1} bits in expectation in subsequent rounds. The net result is that

$$a_j \le 1 + 2^{-t_{j-1}}(t_j + a_{j+1}) \le 1 + \frac{1}{\operatorname{ilog}^{r-j} d}\left(\lceil \operatorname{ilog}^{r-j} d\rceil + 3\right) \le 2 + \frac{4}{\operatorname{ilog}^{r-j} d} \le 3\ .$$

The first two rounds are slightly different, because each player summarily aborts when her input lies in T. In the first round, Alice aborts with probability at most δ. In the second round, conditioned on Alice not aborting, Bob aborts with probability all but $(1-\delta)2^{-t_1}$. The refutation cost of r-round protocols is therefore bounded by

$$\begin{aligned}
\operatorname{rcost}^{\mu}(\mathcal{P}) = a_1 &\le 1 + (1-\delta)t_1 + (1-\delta)\left(1 + (1-\delta)2^{-t_1}(t_2 + a_3)\right) \\
&\le 1 + (1-\delta)(\lceil \operatorname{ilog}^{r-1} \hat{n}\rceil + 1) + (1-\delta)^2 \frac{\lceil \operatorname{ilog}^{r-2} \hat{n}\rceil + 3}{\operatorname{ilog}^{r-2} \hat{n}} \\
&\le 1 + (1-\delta)\operatorname{ilog}^{r-1} \hat{n} + 2(1-\delta) + (1-\delta)^2\left(1 + \frac{4}{\operatorname{ilog}^{r-2} \hat{n}}\right) \\
&\le 1 + (1-\delta)\operatorname{ilog}^{r-1} \hat{n} + 2(1-\delta) + 2(1-\delta)^2 \\
&\le 5 + (1-\delta)\operatorname{ilog}^{r-1} \hat{n}\ .
\end{aligned}$$

◀

▶ **Theorem 24.** *With $n, r, \varepsilon, \delta$ as above, we have $\mathrm{D}^{\mu,(r),\mathrm{ver}}_{\varepsilon,\delta}(\mathrm{EQ}_n) \le (1-\delta)\hat{n} + 3$.*

Proof. We construct a *one-round* protocol achieving the stated verification cost, using S, T, g as in Theorem 23. On input (x, y), Alice aborts if $x \in T$. Otherwise, she sends Bob a prefix of $g(x)$ of length $\min\{n + \lceil \log(1-\delta)\rceil, 2 + \lceil \log((1-\delta)^2/\varepsilon)\rceil$. Bob outputs 0 ("unequal") if (i) Alice aborted, (ii) $y \in T$, or (iii) Alice's prefix does not match that of $g(y)$.

As in the previous proof, this protocol—call it $\mathcal{Q}$—only produces false negatives when inputs lie in T, so that $\operatorname{verr}^{\mu}(\mathcal{Q}) \le \delta$. And as before, we get $\operatorname{rerr}^{\mu}(\mathcal{Q}) = 0$ for small ε and $\operatorname{rerr}^{\mu}(\mathcal{Q}) \le 2^{2\lceil \log(1-\delta)\rceil} \cdot \frac{\varepsilon}{4(1-\delta)^2} \le \varepsilon$ otherwise. As for verification cost, the protocol always sends a bit to indicate abortion (or not), and for all $(x, x) \in S \times S$ the protocol sends at most $\hat{n} + 2$ bits. Thus, $\operatorname{vcost}^{\mu}(\mathcal{Q}) \le 1 + (1-\delta)(\hat{n} + 2) \le (1-\delta)\hat{n} + 3$. ◀

▶ **Theorem 25.** *Let $\mathcal{P}$ be an r-round deterministic protocol for EQ_n. Then, there exists an r-round randomized protocol $\mathcal{Q}$ for EQ_n with $\operatorname{verr}(\mathcal{Q}) = \operatorname{verr}^{\mu}(\mathcal{P})$, $\operatorname{rerr}(\mathcal{Q}) = \operatorname{rerr}^{\mu}(\mathcal{P})$, $\operatorname{rcost}(\mathcal{Q}) = \operatorname{rcost}^{\mu}(\mathcal{P})$, and $\operatorname{vcost}(\mathcal{Q}) = \operatorname{vcost}^{\mu}(\mathcal{P})$.*

Proof. Construct $\mathcal{Q}$ as follows. Alice and Bob use public randomness to generate a uniform bijection $G : \{0,1\}^n \to \{0,1\}^n$. On input (x, y), they run $\mathcal{P}$ on $(G(x), G(y))$. Note that if $x = y$ then $(G(x), G(y))$ is uniform over $\mathrm{EQ}_n^{-1}(1)$, and if $x \ne y$ then $(G(x), G(y))$ is uniform over $\mathrm{EQ}_n^{-1}(0)$. Thus, distributional guarantees for $\mathcal{P}$ under the uniform distribution become worst-case guarantees for $\mathcal{Q}$. ◀

Together with Theorems 23 and 24, this gives upper bounds for randomized protocols.

▶ **Corollary 26.** $\mathrm{R}^{(r),\mathrm{ref}}_{\varepsilon,\delta}(\mathrm{EQ}_n) \le (1-\delta)\operatorname{ilog}^{r-1} \hat{n} + 5$.

▶ **Corollary 27.** $\mathrm{R}^{(r),\mathrm{ver}}_{\varepsilon,\delta}(\mathrm{EQ}_n) \le (1-\delta)\hat{n} + 3$.

C Bounded-Round Communication Lower Bounds for Equality

In this section, we prove all of our communication cost lower bounds on EQ_n. We deal with information cost in the next section. We think of these lower bounds as "combinatorial" (as opposed to "information theoretic"). An important ingredient in some of these combinatorial lower bounds is the *round elimination* technique, which dates back to the work of Miltersen et al. [36].

C.1 Preliminaries

We recall two well-known results from information theory (see, e. g., Cover and Thomas [17]), and state a convenient estimation lemma. The second fact below is one direction of Shannon's source coding theorem. It states that any prefix-free code must have expected length at least the entropy of the source.

▶ **Fact 28** (Kraft Inequality). *Let $S \subseteq \{0,1\}^*$ be a prefix-free set. Then*

$$\sum_{x \in S} 2^{-|x|} \leq 1 \,.$$

▶ **Fact 29** (Source Coding Theorem). *Let X be a random variable taking values in a prefix-free set $S \subseteq \{0,1\}^*$. Then*

$$\mathbb{E}[\,|X|\,] \geq \mathrm{H}(X) \,.$$

▶ **Lemma 30.** *Let X, X' be uniformly distributed over sets $\mathcal{X}, \mathcal{X}'$, respectively, with $\mathcal{X}' \subseteq \mathcal{X}$. Let $f : \mathcal{X} \to \mathbb{R}_+$ be a nonnegative function. Then, we have $\mathbb{E}_{X'}[f(X')] \leq \big(|\mathcal{X}|/|\mathcal{X}'|\big)\, \mathbb{E}_X[f(X)]$.*

Proof. By the nonnegativity of f, we have

$$\mathbb{E}_X[f(X)] = \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} f(x) \geq \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}'} f(x) = \left(\frac{|\mathcal{X}'|}{|\mathcal{X}|}\right) \frac{1}{|\mathcal{X}'|} \sum_{x \in \mathcal{X}'} f(x) = \frac{|\mathcal{X}'|}{|\mathcal{X}|}\, \mathbb{E}_{X'}[f(X')] \,. \blacktriangleleft$$

▶ **Lemma 31.** *For $a \leq 2^{n/2}$, $t \leq \log^* n - 2$, and $x \in \left[\frac{1}{a}, 1\right]$, we have $\operatorname{ilog}^{t-1} n \geq \operatorname{ilog}^t(2^n x) \geq \left(1 - \frac{\log a}{n}\right) \operatorname{ilog}^{t-1} n$.*

Proof. The upper bound is trivial. We prove the lower bound by induction on t. We have $\log(2^n x) = n + \log x \geq n - \log a > \left(1 - \frac{\log a}{n}\right) n$, and the claim holds for $t = 1$. For $t > 1$, we have

$$\begin{aligned}
\operatorname{ilog}^t(2^n x) &\geq \log\left(1 - \frac{\log a}{n}\right) + \log\left(\operatorname{ilog}^{t-2} n\right) && \text{[by induction hypothesis]} \\
&\geq -\frac{2 \log a}{n} + \operatorname{ilog}^{t-1} n && \text{[using } 1 - w \geq 2^{-2w} \text{ for } 0 \leq w \leq 1/2\text{]} \\
&\geq \left(1 - \frac{\log a}{n}\right) \operatorname{ilog}^{t-1} n && \text{[using } \operatorname{ilog}^{t-1} n \geq 2\text{]} \,. \qquad \blacktriangleleft
\end{aligned}$$

C.2 Lower Bounds for Zero-Error Protocols

In this section, we provide nearly exact bounds for zero-error protocols.

▶ **Theorem 32.** *For all $r < \log^* n$ we have $\mathrm{D}^{\mu,(r),\mathrm{ref}}_{0,0}(\mathrm{EQ}_n) \geq \operatorname{ilog}^{r-1} n - 1$.*

To prove this theorem, we must analyze EQUALITY protocols on finite sets of arbitrary size. Given a finite set S, define EQ_S to be the EQUALITY problem, but when $x, y \in S$.

▶ **Theorem 33.** *For all integers $r > 0$, we have $\mathrm{D}^{\mu,(r),\mathrm{ref}}_{0,0}(\mathrm{EQ}_S) \geq \operatorname{ilog}^r |S| - 1$.*

Proof. Assume $\operatorname{ilog}^r |S| > 1$ as otherwise there is nothing to prove. Define m to be the unique real such that $m = \log |S|$. It might be helpful to think of m as an integer, but this is not necessary.

The proof proceeds by induction on r. When $r = 1$, Alice must send her entire input to achieve zero error in a single round. This costs $\lceil m \rceil > \operatorname{ilog}^1 m - 1$ bits, and the theorem

holds. Now, assume $\mathrm{D}^{\mu,(\ell),\mathrm{ref}}_{0,0}(\mathrm{EQ}_T) \geq \operatorname{ilog}^\ell |T| - 1$ for all finite sets T, and let $\mathcal{P}$ be an optimal $(\ell+1)$-round deterministic protocol for EQ_S. We aim to show that $\mathrm{rcost}^\mu(\mathcal{P}) \geq \operatorname{ilog}^{\ell+1} |S| - 1 = \operatorname{ilog}^\ell m - 1$. Let $\mathfrak{m}_1, \ldots, \mathfrak{m}_t$ be the possible messages Alice sends in the first round of $\mathcal{P}$. For $1 \leq i \leq t$, Let A_i denote the set of inputs on which Alice sends $\mathfrak{m}_i$, and let ℓ_i denote the length of $\mathfrak{m}_i$. Assume without loss of generality that $\ell_1 \leq \ell_2 \leq \cdots \leq \ell_t$. Since $\mathcal{P}$ is optimal, we must have $|A_1| \geq |A_2| \geq \cdots \geq |A_t|$: otherwise, we can permute which messages are sent on which sets A_i and reduce the overall cost of the protocol.

We analyze the cost of $\mathcal{P}$ by conditioning on Alice's first message. Under the uniform distribution, Alice sends $\mathfrak{m}_i$ with probability $p_i := |A_i|/2^m$. If $y \notin A_i$, Bob refutes equality and the protocol aborts. Thus, over $x \neq y$ inputs, the probability that Bob aborts is $(|A_i| - 1)/(2^m - 1)$. Furthermore, conditioned on the events that (i) Alice's first message is $\mathfrak{m}_i$ and that (ii) Bob doesn't abort, Alice and Bob's inputs are each uniform over A_i. Thus, the remaining communication is at least $\mathrm{D}^{\mu,(\ell),\mathrm{ref}}_{0,0}(\mathrm{EQ}_{A_i})$.

Fix $\tau := 2/\operatorname{ilog}^{\ell-1} m$. Call the ith message *small* if $p_i \leq \tau$ and *large* otherwise. We bound

$$\begin{aligned}
\mathrm{rcost}^\mu(\mathcal{P}) &= \sum_{1\leq i\leq t} p_i\Big(\ell_i + \frac{|A_i|-1}{2^m-1}\,\mathrm{D}^{\mu,(\ell),\mathrm{ref}}_{0,0}(\mathrm{EQ}_{A_i})\Big)\\
&\geq \sum_{1\leq i\leq t} p_i\Big(-\log p_i + (p_i - 2^{-m})\,\mathrm{D}^{\mu,(\ell),\mathrm{ref}}_{0,0}(\mathrm{EQ}_{A_i})\Big)\\
&\geq \sum_{\text{small } \mathfrak{m}_i} p_i(-\log p_i) + \sum_{\text{large } \mathfrak{m}_i} p_i\Big(-\log p_i + (p_i - 2^{-m})(\operatorname{ilog}^\ell |A_i| - 1)\Big)\\
&\geq \Pr[\text{small message}]\cdot(\operatorname{ilog}^\ell(m) - 1) +\\
&\qquad\qquad \sum_{\text{large } \mathfrak{m}_i} p_i\Big(-\log p_i + p_i \operatorname{ilog}^\ell |A_i| - p_i - 1\Big)\\
&= \Pr[\text{small message}]\cdot(\operatorname{ilog}^\ell(m) - 1) + \sum_{\text{large } \mathfrak{m}_i} p_i f(p_i)\,,
\end{aligned}$$

where we define $f(x) := -\log x + x \operatorname{ilog}^\ell(2^m x) - x - 1$. The first inequality holds by the source coding theorem (Fact 29) and the third inequality holds because $p_i \leq \tau$ for all small messages.

We now claim that $f'(x) > 0$ for all $x \in [\tau, 1]$. We prove this by explicitly calculating the derivative of f. If $x \geq \tau$, then $-1/(x \ln 2) \geq -\operatorname{ilog}^{\ell-1}(m)/(2 \ln 2)$. By Lemma 31, we have

$$\begin{aligned}
f'(x) &= -\frac{1}{x \ln 2} + \operatorname{ilog}^\ell(2^m x) - \frac{1}{(\ln 2)(\ln x \cdot 2^m)\prod_{j=0}^{\ell-2} \ln(\operatorname{ilog}^j x \cdot 2^m)} - 1\\
&\geq -\frac{\operatorname{ilog}^{\ell-1} m}{2 \ln 2} + \operatorname{ilog}^{\ell-1} m - \frac{(\operatorname{ilog}^{\ell-1} m)\operatorname{ilog}^\ell m}{m} - o(1) - 1\\
&= \big(\operatorname{ilog}^{\ell-1} m\big)\left(1 - \frac{1}{2\ln 2}\right) - 1 - o(1) = \Omega(\operatorname{ilog}^{\ell-1} m)\,,
\end{aligned}$$

which proves the claim. It now follows that for large messages, $f(p_i)$ is minimized at $f(\tau)$. Note that

$$\begin{aligned}
f(\tau) &= -\log\tau + \tau \operatorname{ilog}^\ell(2^m\tau) - \tau - 1\\
&\geq \operatorname{ilog}^\ell m - 1 + \frac{2}{\operatorname{ilog}^{\ell-1} m}\operatorname{ilog}^{\ell-1} m\Big(1 - \frac{\operatorname{ilog}^\ell(m) - 1}{m}\Big) - \frac{2}{\operatorname{ilog}^{\ell-1} m} - 1\\
&> \operatorname{ilog}^\ell m - 1\,.
\end{aligned}$$

Plugging this back into our inequality for the cost of $\mathcal{P}$, we get

$$\operatorname{rcost}^{\mu}(\mathcal{P}) \geq \Pr[\text{small message}]\cdot(\operatorname{ilog}^{\ell} m-1)+\Pr[\text{large message}]\cdot(\operatorname{ilog}^{\ell} m-1) = \operatorname{ilog}^{\ell} m-1\,. \blacktriangleleft$$

▶ **Theorem 34.** $\mathrm{D}_{0,0}^{\mu,(r),\mathrm{ver}}(\mathrm{EQ}_n) \geq n$. *Note that this lower bound is independent of* r.

Proof. Let $\mathcal{P}$ be a deterministic zero-error protocol for EQ_n. As the protocol has no error, the communication matrix is partitioned into monochromatic rectangles. In particular, there are 2^n 1-rectangles, since each (x, x) input must map to a different rectangle.[5] Let R_x, T_x, and ℓ_x denote the rectangle consisting of the input pair (x, x), the protocol transcript corresponding to (x, x), and the length of this protocol transcript, respectively. Note that $\{T_x\}$ form a prefix-free coding of $\{0,1\}^n$. By Kraft's inequality, we have $\sum_x 2^{-\ell_x} \leq 1$. Therefore, in expectation $\mathbb{E}[2^{-\ell_x}] \leq 2^{-n}$, and by Jensen's inequality, we get the following.

$$-n \geq \log \mathbb{E}[2^{-\ell_x}] \geq \mathbb{E}[\log(2^{-\ell_x})] = -\mathbb{E}[\ell_x]\,.$$

Multiplying each side of the inequality by -1, we have $\mathbb{E}_x[\ell_x] \geq n$. This is precisely $\operatorname{vcost}^{\mu}(\mathcal{P})$, thus the proof is complete. ◀

▶ **Theorem 35.** $\mathrm{R}_{0,0}^{(r),\mathrm{ver}}(\mathrm{EQ}_n) \geq n$. *As above, this lower bound is independent of* r.

Proof. Let $\mathcal{P}$ be a randomized zero-error protocol for EQ_n. Given any string s, let $\mathcal{P}_s$ denote the deterministic protocol obtained by fixing the public randomness to s. Proceeding along the same lines as in the proof of Theorem 34, we have $\mathbb{E}[\ell_{x,s}] \geq n$, where $\ell_{x,s}$ is the length of the protocol transcript in $\mathcal{P}_s$ on input (x, x). This holds for every $\mathcal{P}_s$, hence $\mathbb{E}_{x,s}[\ell_{x,s}] \geq n$. Therefore, there exists x such that $\mathbb{E}_s[\ell_{x,s}] \geq n$. Recalling the definition of vcost, we have $\operatorname{vcost}(\mathcal{P}) \geq \operatorname{cost}(\mathcal{P}; x, x) = \mathbb{E}_s[\ell_{x,s}] \geq n$, completing the proof. ◀

C.3 Refutation Lower Bounds for Protocols with Two-Sided Error

In this section, we give combinatorial lower bounds on the refutation cost of EQUALITY protocols that admit error. All of the bounds in this section will be asymptotic rather than nearly exact. For this reason, we will strive for simplicity of the proofs at the possible expense of some technical accuracy. For instance, we will often drop ceilings or floors in the mathematical notation. We will also assume that players have the ability to instantly abort a protocol when equality has been refuted. This is easily implemented, as seen in Section C.2 at negligible communication cost. We prefer to avoid the technical machinery needed to express this explicitly.

▶ **Definition 36.** An $\langle n, r, \varepsilon, \delta, c\rangle$-EQUALITY protocol $\mathcal{P}$ is a r-round deterministic protocol with $\operatorname{rerr}^{\mu}(\mathcal{P}) \leq \varepsilon$, $\operatorname{verr}^{\mu}(\mathcal{P}) \leq \delta$, and $\operatorname{rcost}^{\mu}(\mathcal{P}) \leq c$.

For the sake of brevity, we often drop the "EQUALITY" and simply refer to an $\langle n, r, \varepsilon, \delta, c\rangle$-protocol. Our first lemma demonstrates that disallowing false negatives changes the communication complexity very little.

▶ **Lemma 37.** *If there exists a* $\langle n, r, \varepsilon, \delta, c\rangle$-EQUALITY *protocol, then there exists a* $\langle n', r, \varepsilon', 0, c'\rangle$-EQUALITY *protocol, where* $n' = n + \log(1-\delta)$, $\varepsilon' = 2\varepsilon/(1-\delta)^2$, *and* $c' = 2c/(1-\delta)^2$.

[5] If (x, x) and (y, y) were in the same rectangle, then so would (x, y) and (y, x). Thus, the protocol would err on these inputs.

Proof. Let $S = \{x : \mathrm{out}(\mathcal{P}(x,x)) = 0\}$ be the set of inputs on which $\mathcal{P}$ gives a false negative, and let $T = \{0,1\}^n \setminus S$. Since $\mathcal{P}$ has false negative rate δ under the uniform distribution, we have $|T| \geq (1-\delta)2^n = 2^{n'}$.

First create a new EQ_n protocol $\mathcal{P}'$ which works as follows. On input (x,y), Alice aborts and outputs 0 if $x \in S$; otherwise, the players emulate $\mathcal{P}$ and output $\mathrm{out}(\mathcal{P}(x,y))$. Note that $\mathcal{P}'$ makes precisely the same false negatives as in $\mathcal{P}$, and aborting when $x \in S$ can only decrease the false positive rate and the expected communication on inputs in $\mathrm{EQ}_n^{-1}(0)$. Thus, $\mathcal{P}'$ is also a $\langle n, r, \varepsilon, \delta, c\rangle$-protocol.

Next, fix an arbitrary bijection $g : \{0,1\}^{n'} \to T$, and construct an $\mathrm{EQ}_{n'}$ protocol $\mathcal{Q}$ in the following way. On input (X,Y), players emulate $\mathcal{P}'$ on input $(g(X), g(Y))$ and output $\mathrm{out}(\mathcal{P}'(g(X), g(Y)))$. Note that $g(X), g(Y) \in T$, so there are no false negatives. There can be as many false positives as in $\mathcal{P}'$. However, the sample space is smaller ($2^{2n'} - 2^{n'}$ vs $2^{2n} - 2^n$), so the false positive rate can increase. By Lemma 30, the overall error is at most $2\varepsilon/(1-\delta)^2$. Similarly, the communication in $\mathcal{Q}$ on any input (X,Y) is the same as the communication in $\mathcal{P}'$ on input $(g(X), g(Y))$, but since the sample space is smaller (again $2^{2n'} - 2^{n'}$ vs. $2^{2n} - 2^n$), the expected communication can increase. However, the overall increase in communication is at most a factor of $2/(1-\delta)^2$ by Lemma 30. ◀

▶ **Lemma 38** (Combinatorial Round Elimination for EQUALITY). *If there is an $\langle n, r, \varepsilon, 0, c\rangle$-EQUALITY protocol, then there is an $\langle n - 3c - 2, r-1, 12\varepsilon 2^{3c}, 0, 12c2^{3c}\rangle$-EQUALITY protocol.*

Proof. Let $\mathcal{P}$ be a $\langle n, r, \varepsilon, 0, c\rangle$-protocol. Let $Z(x,y) = 1$ if the protocol errs on input (x,y), and let $Z(x,y) = 0$ otherwise. Then we have

$$\mathbb{E}_x\left[\mathbb{E}_{y\neq x}[|\mathcal{P}(x,y)|]\right] \leq c, \quad \text{and} \quad \mathbb{E}_x\left[\mathbb{E}_{y\neq x}[Z(x,y)]\right] \leq \varepsilon\ .$$

Call x good if (1) $\mathbb{E}_{y\neq x}[\mathcal{P}(x,y)|] \leq 3c$, and (2) $\mathbb{E}_{y\neq x}[Z(x,y)] \leq 3\varepsilon$. By two applications of Markov's inequality and a union bound, at least $2^n/3$ x are good. Next, fix Alice's first message m so it is constant over the maximal number of good x. It follows that m is constant over a set A of good x of size $|A| \geq 2^{n-3c-2}$. This induces a $(r-1)$-round protocol $\mathcal{Q}$ for EQ_A. It remains to bound the cost and error of $\mathcal{Q}$. Applying Lemma 30 twice, we have that the cost and error are bounded by (respectively)

$$\begin{aligned}
\mathrm{rcost}^\mu(\mathcal{Q}) &= \mathbb{E}_{x\in A}\left[\mathbb{E}_{y\in A, y\neq x}[|\mathcal{P}(x,y)|]\right] \leq \frac{2^n}{2^{n-3c-2}}\,\mathbb{E}_{x\in A}\left[\mathbb{E}_{y\in\{0,1\}^n, y\neq x}[|\mathcal{P}(x,y)|]\right] \\
&\leq 12c2^{3c}\,, \\
\mathrm{verr}^\mu(\mathcal{Q}) &= \mathbb{E}_{x\in A}\left[\mathbb{E}_{y\in A, y\neq x}[Z(x,y)]\right] \leq \frac{2^n}{2^{n-3c-2}}\,\mathbb{E}_{x\in A}\left[\mathbb{E}_{y\in\{0,1\}^n, y\neq x}[Z(x,y)]\right] \\
&\leq 12\varepsilon 2^{3c}\,.
\end{aligned}$$ ◀

▶ **Corollary 39.** *Let n, j, r, d be integers with $n > d$, d sufficiently large, and $r \geq 1$. Suppose there exists an $\langle n, r, \varepsilon\ell, 0, \ell\rangle$-protocol, where $\ell = \frac{1}{6}\operatorname{ilog}^j d$. Then, there exists an $\langle n - 3\ell - 2, r-1, \varepsilon\ell', 0, \ell'\rangle$-protocol with $\ell' = \frac{1}{6}\operatorname{ilog}^{j-1} d$.*

Proof. This boils down to the following estimations, which are valid for all sufficiently large d.

$$12\ell 2^{3\ell} = 2(\operatorname{ilog}^j d)2^{\frac{1}{2}\operatorname{ilog}^j d} = 2\operatorname{ilog}^j d\sqrt{\operatorname{ilog}^{j-1} d} < \frac{1}{6}\operatorname{ilog}^{j-1} d\,.$$ ◀

▶ **Theorem 40** (Lower Bound for Protocols with False Negatives Disallowed). *Let n be a sufficiently large integer, $\varepsilon < 1/4$ a real, and $r \geq 1$. Fix $\tilde{n} := \min\{n, \log(1/\varepsilon)\}$. Then, $\mathrm{D}^{\mu,(r),\mathrm{ref}}_{\varepsilon,0}(\mathrm{EQ}_n) = \Omega(\operatorname{ilog}^{r-1}\tilde{n})$.*

Proof. In this proof we tacitly assume $\operatorname{ilog}^{r-1}\tilde{n} \geq 100$.

Suppose for the sake of a contradiction that there exists a $\langle n, r, \varepsilon, 0, \frac{1}{6}\operatorname{ilog}^{r-1}\tilde{n}\rangle$-protocol $\mathcal{P}$. Applying Lemma 38 gives an $\langle n - \frac{3}{5}\operatorname{ilog}^{r-1}\tilde{n},\ r-1,\ \frac{\varepsilon}{6}\operatorname{ilog}^{r-2}\tilde{n},\ 0,\ \frac{1}{6}\operatorname{ilog}^{r-2}\tilde{n}\rangle$-protocol $\mathcal{P}'$. Next, applying Corollary 39 repeatedly, a total of $r-2$ times, gives an $\langle n - \frac{3}{5}\sum_{j=1}^{r-1}\operatorname{ilog}^{j}\tilde{n}, 1,\ \frac{\varepsilon}{6}\tilde{n},\ 0,\ \frac{\tilde{n}}{6}\rangle$-protocol. Finally, applying Lemma 38 once more gives an $\langle n - \frac{3}{5}\sum_{j=0}^{r-1}\operatorname{ilog}^{j}\tilde{n},\ 0,\ 2\varepsilon\tilde{n}2^{\tilde{n}/2},\ 0,\ 2\tilde{n}2^{\tilde{n}/2}\rangle$-protocol $\mathcal{Q}$.

Note that since $\mathcal{Q}$ has false negative rate zero, $\mathcal{Q}$ must output 1 with certainty. Thus, $\mathcal{Q}$ errs on all $X \neq Y$ inputs; i. e., $\mathcal{Q}$ has false positive rate 1. On the other hand, $\tilde{n} \leq \log(1/\varepsilon)$, so the false positive rate of $\mathcal{Q}$ is $2\varepsilon\tilde{n}2^{\tilde{n}/6} \leq \sqrt{\varepsilon} < 1/2$. This is a contradiction as long as the problem remains nontrivial.

Since $\operatorname{ilog}^{j}\tilde{n} \geq 100$, we have $\sum_{j=t+1}^{r-1}\operatorname{ilog}^{j}\tilde{n} < \frac{1}{5}\operatorname{ilog}^{t}\tilde{n}$. Also notice that since $\tilde{n} \leq n$, we have $n - \frac{3}{5}\sum_{j=0}^{r-1}\operatorname{ilog}^{j}\tilde{n} > n/5$. Thus, we have a zero-round protocol for $\textsc{eq}_{n'}$ for some $n' = \Omega(n)$ that has false positive rate $< 1/2$ but must output 1 with certainty, a contradiction. ◀

▶ **Theorem 41** (Lower Bound for Protocols with Two-Sided Error)**.** *Let n be a sufficiently large integer, and let ε, δ be reals such that $\delta \leq 1 - 2^{-n/2}$ and $\varepsilon/(1-\delta)^2 < 1/8$. Let $\hat{n}$ be as given in Definition 4. Then, $\mathrm{D}^{\mu,(r),\mathrm{ref}}_{\varepsilon,\delta}(\textsc{eq}_n) = \Omega((1-\delta)^2\operatorname{ilog}^{r-1}\hat{n})$.*

Proof. Fix $d = \min\{n/2, \log((1-\delta)^2/2\varepsilon)\}$, so that $\log d = \Theta(\log\hat{n})$. Suppose, to the contrary, that there exists an $\langle n, r, \varepsilon, \delta, \frac{1}{12}(1-\delta)^2\operatorname{ilog}^{r-1}d\rangle$-protocol $\mathcal{P}$. Since $n + \log(1-\delta) > n/2$, Lemma 37 gives an $\langle n/2, r, 2\varepsilon/(1-\delta)^2, 0, \frac{1}{6}\operatorname{ilog}^{r-1}d\rangle$-protocol. The rest of the proof echoes the proof of Theorem 40. ◀

Next, we prove a combinatorial lower bound for randomized communication complexity.

▶ **Theorem 42.** *Let n be a sufficiently large integer, ε and δ reals such that $\delta < 1 - 2^{1-n/2}$ and $64\varepsilon < (1-\delta)^3$. Then, $\mathrm{R}^{(r),\mathrm{ref}}_{\varepsilon,\delta}(\textsc{eq}_n) = \Omega((1-\delta)^3\operatorname{ilog}^{r-1}\hat{n})$, where $\hat{n}$ is as in Definition 4.*

Proof. Let $\mathcal{P}$ be an r-round randomized protocol with $\mathrm{rerr}(\mathcal{P}) = \varepsilon, \mathrm{verr}(\mathcal{P}) = \delta$, and $\mathrm{rcost}^{\mu}(\mathcal{P}) = c$. Define $z = 1-\delta, \hat{\varepsilon} = 4\varepsilon/(1-\delta)$, and $\hat{c} = 4c/(1-\delta)$. Let $\mathcal{P}_s$ denote the deterministic protocol obtained from $\mathcal{P}$ by setting its random string to s. Call a string s good if (i) $\mathrm{verr}^{\mu}(\mathcal{P}_s) \leq 1 - z/2$, (ii) $\mathrm{rerr}^{\mu}(\mathcal{P}_s) \leq \hat{\varepsilon}$, and (iii) $\mathrm{rcost}^{\mu}(\mathcal{P}_s) \leq \hat{c}$. Applying a Markov argument to each of these three conditions, we see that

$$\Pr[s \text{ is bad}] < \frac{1-z}{1-z/2} + \frac{z}{4} + \frac{z}{4} < 1\,,$$

where we used $(1-z)/(1-z/2) < 1 - z/2$. Thus there exists a good string s. Note that $\mathcal{P}_s$ is a $[n, r, \hat{\varepsilon}, \hat{\delta}, \hat{c}]$-protocol, and by Theorem 41, $\hat{c} = \Omega((1-\delta)^2\operatorname{ilog}^{r-1}\hat{n})$. Therefore, $c = \Omega((1-\delta)^3\operatorname{ilog}^{r-1}\hat{n})$. ◀

C.4 Verification Lower Bounds for Protocols with Two-Sided Error

▶ **Theorem 43.** $\mathrm{D}^{\mu,(r),\mathrm{ver}}_{\varepsilon,\delta}(\textsc{eq}_n) \geq (1-\delta)(\hat{n}-1)$, *where $\hat{n}$ is as in Definition 4.*

Proof. Fix a deterministic protocol $\mathcal{P}$ achieving $\mathrm{rerr}^{\mu}(\mathcal{P}) = \varepsilon$ and $\mathrm{verr}^{\mu}(\mathcal{P}) = \delta$. This protocol naturally partitions the communication matrix for $\textsc{eq}_n$ into combinatorial rectangles. Let $R_1, \ldots, R_c$ be the rectangles on which $\mathcal{P}$ outputs 1. Let s_i denote the number of (x,x) inputs in R_i. Since $\mathcal{P}$ has false negative rate δ, we have $\sum_i s_i = 2^n(1-\delta)$. Let $p_i = s_i/2^n$ and $q_i = p_i/(1-\delta)$. Notice that p_i is the probability that $(x,x) \in R_i$ for a uniformly chosen x. Similarly, q_i is the probability that $(x,x) \in R_i$ conditioned on $\mathcal{P}$ verifying equality on

(x,x). We now analyze the false positive rate. Recall that there are $2^{2n}-2^n$ total $x \neq y$ inputs. It is easy to see that R_i contains at least $s_i^2 - s_i$ false positives. Therefore, we have

$$\varepsilon \geq \frac{1}{2^{2n}-2^n}\sum_{i=1}^{c}(s_i^2 - s_i) = \sum_{i=1}^{c}\frac{s_i(s_i-1)}{2^n(2^n-1)} \geq \sum_{i=1}^{c} p_i(p_i - 2^{-n}) = -2^{-n}(1-\delta) + \sum_{i=1}^{c} p_i^2 \,.$$

Rearranging terms and noting that $q_i = p_i/(1-\delta)$, we have

$$\mathbb{E}[q_i] = \sum_{i=1}^{c} q_i^2 = \frac{1}{(1-\delta)^2}\sum_{i=1}^{c} p_i^2 \leq \frac{1}{(1-\delta)^2}\left(\varepsilon + 2^{-n}(1-\delta)\right) = \frac{\varepsilon}{(1-\delta)^2} + \frac{2^{-n}}{(1-\delta)} \leq 2\cdot 2^{-\hat{n}} \,.$$

Next, we analyze the verification cost of $\mathcal{P}$. Let ℓ_i denote the length of the protocol transcript for inputs in the rectangle R_i. Observe that the transcripts $\mathcal{P}(x,x)$ with $\mathrm{out}(\mathcal{P}(x,x)) = 1$ give a prefix-free encoding of the set of rectangles $\{R_1, \ldots, R_c\}$. Therefore,

$$\begin{aligned}\mathrm{vcost}^{\mu}(\mathcal{P}) &= \sum_{x\in\{0,1\}^n} \frac{|\mathcal{P}(x,x)|}{2^n} \geq \sum_{i=1}^{c} p_i\ell_i = (1-\delta)\sum_{i=1}^{c} q_i\ell_i \geq (1-\delta)\sum_{i=1}^{c} q_i(-\log q_i) \\ &= -(1-\delta)\,\mathbb{E}[\log q_i] \geq -(1-\delta)\log \mathbb{E}[q_i] \geq -(1-\delta)(-\hat{n}+1) = (1-\delta)(\hat{n}-1) \,,\end{aligned}$$

where the second inequality is from the source coding theorem (Fact 29) and the third is from Jensen's inequality. ◀

▶ **Theorem 44.** $\mathrm{R}^{(r),\mathrm{ver}}_{\varepsilon,\delta}(\mathrm{EQ}_n) > \frac{1}{8}(1-\delta)^2(\hat{n} + \log(1-\delta) - 5)$.

Proof. Suppose there exists a randomized protocol $\mathcal{P}$ with $\mathrm{rerr}(\mathcal{P}) \leq \varepsilon$, $\mathrm{verr}(\mathcal{P}) \leq \delta$, and $\mathrm{vcost}(\mathcal{P}) \leq m$. For a string s, let $\mathcal{P}_s$ denote the deterministic protocol obtained from $\mathcal{P}$ by fixing the public randomness to s. By the cost and error guarantees of $\mathcal{P}$, for all $(x,y) \in \mathrm{EQ}_n^{-1}(1)$ we have $\mathbb{E}_s\left[\mathrm{cost}(\mathcal{P}_s; x, y)\right] \leq m$ and $\mathbb{E}_s\left[\Pr[\mathrm{out}(\mathcal{P}_s(x,y)) = 0]\right] \leq \delta$, while for $(x,y) \in \mathrm{EQ}^{-1}(0)$ we have $\mathbb{E}_s\left[\Pr[\mathrm{out}(\mathcal{P}_s(x,y)) = 1]\right] \leq \varepsilon$. In particular, letting $(X,Y) \sim \mu$, we have

$$\begin{aligned}\mathbb{E}_{s,X,Y}\left[\Pr[\mathrm{out}(\mathcal{P}_s(X,Y)) = 1 \mid X \neq Y]\right] &\leq \varepsilon \,, \\ \mathbb{E}_{s,X,Y}\left[\Pr[\mathrm{out}(\mathcal{P}_s(X,Y) = 0 \mid X = Y]\right] &\leq \delta \,, \\ \mathbb{E}_{s,X,Y}\left[\mathrm{cost}(\mathcal{P}_s; X, Y) \mid X = Y\right] &\leq m \,.\end{aligned}$$

Define $z = 1-\delta$, $\hat{\varepsilon} = 4\varepsilon/(1-\delta)$, $\hat{\delta} = 1 - z/2$, and $\hat{m} = 4m/(1-\delta)$. Call a string s good if (i) $\mathrm{verr}(\mathcal{P}_s) \leq 1 - z/2$, (ii) $\mathrm{rerr}(\mathcal{P}_s) \leq \hat{\varepsilon}$, and (iii) $\mathrm{vcost}^{\mu}(\mathcal{P}) \leq \hat{m}$. Applying a Markov argument to each condition,

$$\Pr[s \text{ is bad}] < \frac{1-z}{1-z/2} + \frac{z}{4} + \frac{z}{4} < 1 \,,$$

where we used $(1-z)/(1-z/2) < 1 - z/2$. Thus, there exists a good string s. Note that $\mathcal{P}_s$ is a deterministic $(\hat{\varepsilon}, \hat{\delta})$-error EQ_n protocol. Using Definition 4 to figure the new effective instance size and applying Theorem 43, we obtain

$$\frac{4m}{1-\delta} \geq \mathrm{vcost}^{\mu}(\mathcal{P}_s) \geq \frac{z}{2}\left(\min\left\{n + \log(z/2), \log\frac{z(z/2)^2}{4\varepsilon}\right\} - 1\right) \geq \frac{z}{2}(\hat{n} + \log z - 5) \,.$$

The proof is completed by rearranging the above inequality and substituting $z = 1-\delta$. ◀

The analysis in the above proof is very loose when δ is bounded away from 1. In particular, when there are no false negatives (i. e., when $\delta = 0$), we are able to show that $\mathrm{R}^{(r),\mathrm{ver}}_{\varepsilon,0} \geq c\hat{n}$ for every constant $c < 1$.

D Main Theorem: Bounded-Round Information Complexity of Equality

In this section we prove Theorem 6, which we think of as the most important result of this paper. We wish to lower bound the bounded-round information complexity of EQUALITY with respect to the uniform distribution. Recall that we are concerned chiefly with protocols that achieve very low refutation error, though they may have rather high verification error. We will prove our lower bound by proving a round elimination lemma for EQ_n that targets *information* cost, and then applying this lemma repeatedly.

This proof has much more technical complexity than our earlier lower bound proofs. Let us see why. There are two main technical difficulties and they arise, ultimately, from the same source: the inability to use (the easy direction of) Yao's minimax lemma. When proving a lower bound on *communication* cost, Yao's lemma allows us to fix the random string used by any purported protocol, which immediately moves us into the clean world of deterministic protocols. This hammer is unavailable to us when working with *information* cost. The most we can do is to "average away" the public randomness. We then have to deal with (private coin) randomized protocols the entire way through the round elimination argument. As a result, our intermediate protocols, obtained by eliminating some rounds of our original protocol, do not obey straightforward cost and error guarantees. This is the first technical difficulty, and our solution to it leads us to the concept of a "kernel" in Definition 45 below.

The second technical difficulty is that we are unable to switch to the simpler case of zero verification error like we did in the proof of Theorem 5, Parts (9) and (10). Therefore, all our intermediate protocols continue to have verification error. Since errors scale up with each round elimination, and the verification error starts out high, we cannot afford even a constant-factor scaling. We must play very delicately with our error parameters, which leads us to the somewhat complicated parametrization seen in Definition 46 below.

D.1 The Round Elimination Argument

▶ **Definition 45** (Kernel)**.** Let p and q be probability distributions on $\{0,1\}^n$, let $S \subseteq \{0,1\}^n$, and let $\ell \geq 0$ be a real number. The triple (p, q, S) is defined to be an *ℓ-kernel* if the following properties hold.

- [K1] $\text{H}(p) \geq n - \ell$ and $\text{H}(q) \geq n - \ell$.
- [K2] $p(S) \geq 2^{-\ell}$ and $q(S) \geq \frac{1}{2}$.
- [K3] For all $x \in S$ we have $q(x) \geq 2^{-n-\ell}$.

▶ **Definition 46** (Parametrized Protocols)**.** Suppose we have an integer $r \geq 1$, and nonnegative reals ℓ, a, b, and c. A protocol $\mathcal{P}$ for EQ_n is defined to be an $[r, \ell, a, b, c]$-protocol if there exists an ℓ-kernel (p, q, S) such that the following properties hold.

- [P1] The protocol $\mathcal{P}$ is private-coin and uses r rounds, with Alice speaking in the first round.
- [P2] We have $\text{err}^{p\otimes q|S\times S}(\mathcal{P}) = \Pr_{(X,Y)\sim p\otimes q}[\text{out}(\mathcal{P}(X,Y)) \neq \text{EQ}_n(X,Y) \mid (X,Y) \in S \times S] \leq 2^{-a}$.
- [P3] We have $\text{verr}^{p\otimes\xi|S\times S}(\mathcal{P}) = \Pr_{X\sim p}[\text{out}(\mathcal{P}(X,X)) = 0 \mid X \in S] \leq 1 - 2^{-b}$.
- [P4] We have $\text{icost}^{p\otimes q}(\mathcal{P}) \leq c$.

We alert the reader to the fact that [P2] considers overall error, and not refutation error. We encourage the reader to take a careful look at [P3] and verify the equality claimed therein.

It is straightforward, once one revisits Definition 1 and recalls that ξ denotes the uniform distribution on $\{0,1\}^n$.

Since we have a number of parameters at play, it is worth recording the following simple observation.

▶ **Fact 47.** *Suppose that $\ell' \geq \ell, c' \geq c, a' \leq a$, and $b' \geq b$. Then every ℓ-kernel is also an ℓ'-kernel, and every $[r,\ell,a,b,c]$-protocol is also an $[r,\ell',a',b',c']$-protocol.* ◀

▶ **Theorem 48** (Information-Theoretic Round Elimination for EQUALITY)**.** *If there exists an $[r,\ell,a,b,c]$-protocol with $r \geq 1$ and $c \geq 4$, then there exists an $[r-1,\ell',a',b',c']$-protocol, where*

$$\ell' := (c+\ell)2^{\ell+2b+7}\,, \qquad a' := a-(c+\ell)2^{\ell+2b+8}\,,$$
$$b' := b+2\,, \qquad c' := (c+2)2^{\ell+2b+6}\,.$$

Proof. Let $\mathcal{P}$ be an $[r,\ell,a,b,c]$-protocol, and let (p,q,S) be an ℓ-kernel satisfying the conditions in Definition 46. Assume WLOG that the each message in $\mathcal{P}$ is generated using a fresh random string. Let $X \sim p$ and $Y \sim q$ be independent random variables denoting an input to $\mathcal{P}$. Let $M_1,\ldots,M_r$ be random variables denoting the messages sent in $\mathcal{P}$ on input (X,Y), with M_j being the jth message; note that these variables depend on X,Y, and the random strings used by the players. We then have

$$c \geq \mathrm{icost}^{p\otimes q}(\mathcal{P}) = \mathrm{I}(XY : M_1M_2\ldots M_r) = \mathrm{I}(X : M_1) + \mathrm{I}(XY : M_2\ldots M_r \mid M_1)\,, \tag{15}$$

where the final step uses the chain rule for mutual information, and the fact that M_1 and Y are independent. In particular, we have $\mathrm{I}(X : M_1) \leq c$, and so $\mathrm{H}(X \mid M_1) = \mathrm{H}(X) - \mathrm{I}(X : M_1) \geq n-\ell-c$. By Lemma 19,

$$\mathrm{H}(X \mid M_1, X \in S) \geq n - \frac{\ell+c+1}{p(S)} \geq n-(\ell+c+1)2^{\ell}\,. \tag{16}$$

Let $\mathcal{M}$ be the set of messages that Alice sends with positive probability as her first message in $\mathcal{P}$, given the random input X, i. e., $\mathcal{M} := \{\mathfrak{m} : \Pr[M_1 = \mathfrak{m}] > 0\}$. Consider a particular message $\mathfrak{m} \in \mathcal{M}$. Let $\mathcal{P}'_{\mathfrak{m}}$ denote the following protocol for EQ_n. The players simulate $\mathcal{P}$ on their input, except that Alice is assumed to have sent $\mathfrak{m}$ as her first message. As a result, $\mathcal{P}'_{\mathfrak{m}}$ has $r-1$ rounds and Bob is the player to send the first message in $\mathcal{P}'_{\mathfrak{m}}$. Let $\pi_{\mathfrak{m}}$ and q' be the distributions of $(X \mid M_1 = \mathfrak{m} \wedge X \in S)$ and $(Y \mid Y \in S)$, respectively.

Observe that $\mathrm{icost}^{\pi_{\mathfrak{m}}\otimes q'}(\mathcal{P}'_{\mathfrak{m}}) = \mathrm{I}(XY : M_2\ldots M_r \mid M_1 = \mathfrak{m} \wedge (X,Y) \in S\times S)$. Letting L denote a random first message distributed identically to M_1, we now get

$$\begin{aligned}\mathbb{E}_L\left[\mathrm{icost}^{\pi_L\otimes q'}(\mathcal{P}'_L)\right] &= \mathrm{I}(XY : M_2\ldots M_r \mid M_1, (X,Y) \in S\times S)\\ &\leq \frac{\mathrm{I}(XY : M_2\ldots M_r \mid M_1)+1}{p(S)q(S)} \leq (c+1)2^{\ell+1}\,,\end{aligned} \tag{17}$$

where the first inequality uses Lemma 18 and the second inequality uses (15) and Property [K2]. Examining Properties [P2] and [P3], we obtain

$$\mathbb{E}_L\left[\mathrm{err}^{\pi_L\otimes q'}(\mathcal{P}'_L)\right] = \mathrm{err}^{p\otimes q|S\times S}(\mathcal{P}) \leq 2^{-a}\,, \tag{18}$$
$$\mathbb{E}_L\left[\mathrm{verr}^{\pi_L\otimes \xi}(\mathcal{P}'_L)\right] = \mathrm{verr}^{p\otimes \xi|S\times S}(\mathcal{P}) \leq 1-2^{-b}\,. \tag{19}$$

▶ **Definition 49** (Good Message)**.** A message $\mathfrak{m} \in \mathcal{M}$ is said to be *good* if the following properties hold:

[G1] $\mathrm{H}(\pi_{\mathfrak{m}}) = \mathrm{H}(X \mid M_1 = \mathfrak{m} \wedge X \in S) \geq n - (\ell + c + 1)2^{\ell+b+3}$,
[G2] $\mathrm{icost}^{\pi_{\mathfrak{m}} \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) \leq 2^{\ell+b+4}(c+1)$,
[G3] $\mathrm{err}^{\pi_{\mathfrak{m}} \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) \leq 2^{-a+b+3}$,
[G4] $\mathrm{verr}^{\pi_{\mathfrak{m}} \otimes \xi}(\mathcal{P}'_{\mathfrak{m}}) \leq 1 - 2^{-b-1}$.

Notice that for all $\mathfrak{m} \in \mathcal{M}$ we have $\mathrm{H}(X \mid M_1 = \mathfrak{m}, X \in S) \leq n$. Hence, viewing (16), (17), (18) and (19) as upper bounds on the expected values of certain nonnegative functions of L, we may apply Markov's inequality to these four conditions and conclude that

$$\Pr[L \text{ is good}] \geq 1 - 2^{-b-3} - 2^{-b-3} - 2^{-b-3} - \frac{1 - 2^{-b}}{1 - 2^{-b-1}} \geq 2^{-b-1} - 3 \cdot 2^{-b-3} > 0\,.$$

Thus, there exists a good message. *From now on, we fix $\mathfrak{m}$ to be such a good message.*

We may rewrite the left-hand side of [G4] as $\mathbb{E}_{Z \sim \pi_{\mathfrak{m}}}[\Pr[\mathrm{out}(\mathcal{P}'_{\mathfrak{m}}(Z, Z)) = 0]]$. So if we define the set $T := \left\{x \in S : \Pr[\mathrm{out}(\mathcal{P}'_{\mathfrak{m}}(x, x)) = 0] \leq 1 - 2^{-b-2}\right\}$ and apply Markov's inequality again, we obtain

$$\pi_{\mathfrak{m}}(T) \geq 1 - \frac{1 - 2^{-b-1}}{1 - 2^{-b-2}} \geq 2^{-b-2}\,. \tag{20}$$

Defining the distribution $p' := \pi_{\mathfrak{m}} \mid T$ and the set $S' := \{x \in T : p'(x) \geq 2^{-n-\ell'}\}$, we now make two claims.

Claim 1: The triple (q', p', S') is an ℓ'-kernel.
Claim 2: We have $\mathrm{err}^{p' \otimes q' \mid S' \times S'}(\mathcal{P}'_{\mathfrak{m}}) \leq 2^{-a'}$, $\mathrm{verr}^{q' \otimes \xi \mid S' \times S'}(\mathcal{P}'_{\mathfrak{m}}) \leq 1 - 2^{-b'}$, and $\mathrm{icost}^{p' \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) \leq c'$.

Notice that these claims essentially say that $\mathcal{P}'_{\mathfrak{m}}$ has all the properties listed in Definition 46, except that Bob starts $\mathcal{P}'_{\mathfrak{m}}$. Interchanging the roles of Alice and Bob in $\mathcal{P}'_{\mathfrak{m}}$ gives us the desired $[r-1, \ell', a', b', c']$-protocol, which completes the proof of the theorem.

It remains to prove the above claims. We start with Claim 1. Starting with the lower bound on $\mathrm{H}(\pi_{\mathfrak{m}})$ given by Property [G1] of the good message $\mathfrak{m}$, and using Lemma 19 followed by (20), we obtain

$$\mathrm{H}(p') = \mathrm{H}(\pi_{\mathfrak{m}} \mid T) \geq n - \frac{(c + \ell + 1)2^{\ell+b+3} + 1}{\pi_{\mathfrak{m}}(T)} \geq n - (c + \ell + 2)2^{\ell+2b+5} \geq n - \ell'\,. \tag{21}$$

We may lower bound $\mathrm{H}(q')$ using Properties [K1] and [K2] for (p, q, S) and applying Lemma 19. We have

$$\mathrm{H}(q') = \mathrm{H}(Y \mid Y \in S) \geq n - \frac{\ell + 1}{q(S)} \geq n - 2(\ell + 1) \geq n - \ell'\,.$$

Thus, (q', p', S') satisfies Property [K1] for an ℓ'-kernel. It is immediate that it also satisfies Property [K3]: by definition, for all $x \in S'$, we have $p'(x) \geq 2^{-n-\ell'}$.

It remains to verify Property [K2], which entails showing that $p'(S') \geq \frac{1}{2}$ and that $q'(S') \geq 2^{-\ell'}$. We can lower bound $p'(S')$ as follows:

$$p'(S') = 1 - \sum_{x \in \{0,1\}^n \setminus S'} p'(x) = 1 - \sum_{\substack{x \in \{0,1\}^n \\ p'(x) < 2^{-n-\ell'}}} p'(x) \geq 1 - 2^{-\ell'} \geq \frac{1}{2}\,. \tag{22}$$

To prove the second inequality, we first derive a lower bound on $\mathrm{H}(p' \mid S')$, thence on $|S'|$, and finally on $q'(S')$. We already showed that $\mathrm{H}(p') \geq n - (c + \ell + 2)2^{\ell+2b+5}$, at (21). By Lemma 19 and (22), we get

$$\mathrm{H}(p' \mid S') \geq n - \frac{(c + \ell + 2)2^{\ell+2b+5} + 1}{p'(S')} \geq n - \left((c + \ell + 2)2^{\ell+2b+6} + 2\right) \geq n - (c + \ell + 4)2^{\ell+2b+6}\,,$$

and so $|S'| \geq 2^{n-(c+\ell+4)2^{\ell+2b+6}}$. Since $q' = q \mid S$ and $S' \subseteq S$, we have

$$q'(S') \geq q(S') \geq |S'| \min_{y \in S'} q(y) \geq |S'| \min_{y \in S} q(y) \geq 2^{n-(c+\ell+4)2^{\ell+2b+6}} 2^{-n-\ell} = 2^{-\ell-(c+\ell+4)2^{\ell+2b+6}},$$

where the final inequality uses Property [K3]. Recalling the definition of ℓ' and applying a crude estimate (using the bound $c \geq 4$), we get $q'(S') \geq 2^{-\ell'}$. This finishes the proof of Claim 1.

We now prove Claim 2. Of the three bounds we need to prove, the verification error bound is the easiest. Recalling how T was defined, and noting that $S' \subseteq T$, we immediately obtain

$$\mathrm{verr}^{q' \otimes \xi | S' \times S'}(\mathcal{P}'_{\mathfrak{m}}) = \mathbb{E}_{Y' \sim q'}[\Pr[\mathrm{out}(\mathcal{P}'_{\mathfrak{m}}(Y', Y')) = 0 \mid Y' \in S']] \leq 1 - 2^{-b-2}.$$

To establish the overall error bound, we use

$$\mathrm{err}^{p' \otimes q' | S' \times S'}(\mathcal{P}'_{\mathfrak{m}}) \leq \frac{\mathrm{err}^{p' \otimes q'}(\mathcal{P}'_{\mathfrak{m}})}{p'(S')q'(S')} \leq \frac{\mathrm{err}^{\pi_{\mathfrak{m}} \otimes q'}(\mathcal{P}'_{\mathfrak{m}})}{\pi_{\mathfrak{m}}(T)p'(S')q'(S')} \leq \frac{2^{-a+b+3}}{2^{-b-2} \cdot \frac{1}{2} \cdot 2^{-\ell'}} \tag{23}$$

$$= 2^{-a+2b+6+(c+\ell)2^{\ell+2b+7}} \leq 2^{-a+(c+\ell)2^{\ell+2b+8}}, \tag{24}$$

where the final inequality in (23) follows from Property [K2] for an ℓ'-kernel and Property [G3], and (24) just uses a crude estimate (this time $c \geq 1$ suffices). The last thing remaining is to establish the information cost bound in Claim 2. We do this as follows.

$$\begin{aligned}
\mathrm{icost}^{p' \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) &= \mathrm{I}(XY : M_2 \dots M_r \mid M_1 = \mathfrak{m} \wedge X \in T \wedge Y \in S) \\
&\leq \frac{\mathrm{I}(XY : M_2 \dots M_r \mid M_1 = \mathfrak{m} \wedge (X,Y) \in S \times S) + 1}{\Pr[X \in T \mid M_1 = \mathfrak{m} \wedge (X,Y) \in S \times S]} && (25) \\
&= \frac{\mathrm{icost}^{\pi_{\mathfrak{m}} \otimes q'}(\mathcal{P}'_{\mathfrak{m}}) + 1}{\pi_{\mathfrak{m}}(T)} && (26) \\
&\leq \frac{2^{b+\ell+4}(c+1)+1}{2^{-b-2}} \leq (c+2)2^{\ell+2b+6}, && (27)
\end{aligned}$$

where (25) uses Lemma 18, (26) uses the independence of X and Y and (27) uses Property [G2] and Eq. (20).

This completes the proof of Claim 2 and, with it, the proof of the theorem. ◀

The following easy corollary of Theorem 48 will be useful shortly.

▶ **Corollary 50.** *Let $\tilde{n}, j, r \in \mathbb{N}$ and $a, b \in \mathbb{R}$ with $\tilde{n}$ sufficiently large, $j \geq 1$, $r \geq 1$, and $b \geq 0$. Suppose there exists an $[r, \ell, a-\ell, b, \ell]$-protocol, with $b \leq \ell = \frac{1}{8}\operatorname{ilog}^j \tilde{n}$. Then there exists an $[r-1, \ell', a-\ell', b+2, \ell']$-protocol with $b+2 \leq \ell' = (\operatorname{ilog}^{j-1} \tilde{n})^{1/2} \leq \frac{1}{8}\operatorname{ilog}^{j-1} \tilde{n}$.*

Proof. This simply boils down to the following estimation, which is valid for all sufficiently large $\tilde{n}$:

$$(\ell+\ell)2^{\ell+2b+8} = 2^7(\operatorname{ilog}^j \tilde{n})2^{(3/8)\operatorname{ilog}^j \tilde{n}} = 2^7(\operatorname{ilog}^{j-1} \tilde{n})^{3/8} \log(\operatorname{ilog}^{j-1} \tilde{n}) \leq (\operatorname{ilog}^{j-1} \tilde{n})^{1/2}. \blacktriangleleft$$

D.2 Finishing the Proof

We are now ready to state and prove the main lower bound on protocols with two-sided error.

▶ **Theorem 51** (Restatement of Main Theorem). *Let $\tilde{n} = \min\{n + \log(1-\delta), \log((1-\delta)/\varepsilon)\}$. Suppose $\delta \leq 1 - 8(\operatorname{ilog}^{r-2} \tilde{n})^{-1/8}$. Then we have $\mathrm{IC}^{\mu,(r)}_{\varepsilon,\delta}(\mathrm{EQ}_n) = \Omega((1-\delta)^3 \operatorname{ilog}^{r-1} \tilde{n})$.*

Proof. We may assume that $r \le \log^* \tilde{n}$, for otherwise there is nothing to prove. The slight difference between $\tilde{n}$ above and $\hat{n}$, as in Definition 4, is insignificant and can be absorbed by the $\Omega(\cdot)$ notation.

Suppose, to the contrary, that there exists an r-round randomized protocol $\mathcal{P}^*$ for EQ_n, with $\mathrm{rerr}^\mu(\mathcal{P}^*) \le \varepsilon$, $\mathrm{verr}^\mu(\mathcal{P}^*) \le \delta$ and $\mathrm{icost}^\mu(\mathcal{P}^*) \le 2^{-16}(1-\delta)^3 \operatorname{ilog}^{r-1} \tilde{n}$. Recall that we denote the uniform distribution on $\{0,1\}^n$ by ξ and that $\mu = \xi \otimes \xi$. We have

$$\mathrm{err}^\mu(\mathcal{P}^*) = (1-2^{-n})\,\mathrm{rerr}^\mu(\mathcal{P}^*) + 2^{-n}\,\mathrm{verr}^\mu(\mathcal{P}^*) \le \varepsilon + 2^{-n}(\delta - \varepsilon) \le \varepsilon + 2^{-n}\,.$$

Let $\mathcal{P}^*_s$ be the private-coin protocol for EQ_n obtained from $\mathcal{P}^*$ by fixing the public random string of $\mathcal{P}^*$ to be s. We have $\mathbb{E}_s[\mathrm{err}^\mu(\mathcal{P}^*_s)] \le \varepsilon + 2^{-n}$, $\mathbb{E}_s[\mathrm{verr}^\mu(\mathcal{P}^*_s)] \le \delta$, and $\mathbb{E}_s[\mathrm{icost}(\mathcal{P}^*_s)] \le 2^{-16}(1-\delta)^3 \operatorname{ilog}^{r-1} \tilde{n}$. By Markov's inequality, there exists s such that $\mathcal{P}^*_s$ simultaneously has $\mathrm{err}^\mu(\mathcal{P}^*_s) \le 4(\varepsilon + 2^{-n})/(1-\delta)$, $\mathrm{verr}^\mu(\mathcal{P}^*_s) \le (1+\delta)/2$, and $\mathrm{icost}(\mathcal{P}^*_s) \le 2^{-14}(1-\delta)^2 \operatorname{ilog}^{r-1} \tilde{n}$: this is because

$$1 - \frac{1-\delta}{4} - \frac{2\delta}{1+\delta} - \frac{1-\delta}{4} = \frac{(1-\delta)^2}{2(1+\delta)} > 0\,.$$

Let $\mathcal{P} = \mathcal{P}^*_s$ for this s. Then $(\xi, \xi, \{0,1\}^n)$ is a 0-kernel and $\mathcal{P}$ is an $[r,\, 0,\, \log \frac{1-\delta}{4(\varepsilon+2^{-n})},\, \log \frac{2}{1-\delta},\, 2^{-14}(1-\delta)^2 \operatorname{ilog}^{r-1} \tilde{n}]$-protocol. Recalling Fact 47 and using $\log \frac{1-\delta}{\varepsilon+2^{-n}} \ge \tilde{n} - 1$, we see that

$$\mathcal{P} \text{ is an } \left[r,\, 0,\, \tilde{n}-3,\, \log \tfrac{1}{1-\delta} + 1,\, 2^{-14}(1-\delta)^2 \operatorname{ilog}^{r-1} \tilde{n}\right]\text{-protocol.}$$

Put $\ell_j := \frac{1}{8} \operatorname{ilog}^j \tilde{n}$ for $j \in \mathbb{N}$. Applying round elimination (Theorem 48) to $\mathcal{P}$ and weakening the resulting parameters (using Fact 47) gives us an $[r-1, \ell_{r-1}, \tilde{n} - \ell_{r-1}, \log \frac{1}{1-\delta} + 3, \ell_{r-1}]$-protocol $\mathcal{P}'$.

The upper bound on δ gives us $\log \frac{1}{1-\delta} + 3 \le \ell_{r-1}$, and so the conditions for Corollary 50 apply. Starting with $\mathcal{P}'$ and applying that corollary repeatedly, each time using the looser estimate on ℓ' in that corollary, we obtain a sequence of protocols with successively fewer rounds. Eventually we reach a $[1, \ell_1, \tilde{n} - \ell_1, \log \frac{1}{1-\delta} + 2(r-1) + 1, \ell_1]$-protocol. Applying Theorem 48 one more time, and using the tighter estimate on ℓ' this time, we get a $[0,\, \tilde{n}^{1/2},\, \tilde{n} - \tilde{n}^{1/2},\, \log \frac{1}{1-\delta} + 2r + 1,\, \tilde{n}^{1/2}]$-protocol $\mathcal{Q}$. Weakening parameters again, we see that $\mathcal{Q}$ is a $[0,\, \tilde{n}^{1/2},\, \frac{1}{2}\tilde{n},\, \frac{1}{3}\log \tilde{n},\, \tilde{n}^{1/2}]$-protocol. Let (p, q, S) be the $\tilde{n}^{1/2}$-kernel for $\mathcal{Q}$. By Property [K1], we have $\mathrm{H}(q) \ge n - \tilde{n}^{1/2}$. Using Lemma 19 and Property [K2], we then have

$$\mathrm{H}(q \mid S) \ge n - \frac{\tilde{n}^{1/2} + 1}{q(S)} \ge n - (2\tilde{n}^{1/2} + 2)\,. \tag{28}$$

Since $\mathcal{Q}$ involves no communication, it must behave identically on any two input distributions that have the same marginal on Alice's input. In particular, this gives us the following crucial equation:

$$\Pr_{X \sim p}[\mathrm{out}(\mathcal{Q}(X,X)) = 1 \mid X \in S] = \Pr_{(X,Y) \sim p \otimes q}[\mathrm{out}(\mathcal{Q}(X,Y)) = 1 \mid (X,Y) \in S \times S]\,. \tag{29}$$

Let α denote the above probability. Considering the left-hand side of (29), we have

$$\alpha = 1 - \mathrm{verr}^{p \otimes \xi | S \times S}(\mathcal{Q}) \ge 2^{-\frac{1}{3}\log \tilde{n}} = \tilde{n}^{-1/3}\,. \tag{30}$$

On the other hand, whenever $\mathcal{Q}$ outputs 1 on an input (x, y), then either $x = y$ or $\mathcal{Q}$ errs on

(x, y). Therefore, considering the right-hand side of (29), we have

$$\begin{aligned}\alpha &\leq \Pr_{(X,Y)\sim p\otimes q}[X = Y \mid (X,Y) \in S \times S] + \\ &\qquad \Pr_{(X,Y)\sim p\otimes q}[\mathrm{out}(\mathcal{P}(X,Y)) \neq \mathrm{EQ}_n(X,Y) \mid (X,Y) \in S \times S] \qquad (31)\\ &\leq \max_{x\in S} \Pr_{Y\sim q|S}[Y = x] + \mathrm{err}^{p\otimes q|S\times S}(\mathcal{Q}) \\ &\leq \frac{2\tilde{n}^{1/2}+3}{n} + 2^{-\frac{1}{2}\tilde{n}} \qquad (32)\\ &\leq 2\tilde{n}^{-1/2} + 3\tilde{n}^{-1} + 2^{-\frac{1}{2}\tilde{n}}\,, \qquad (33)\end{aligned}$$

where (32) follows from (28) by applying Lemma 20, and (33) uses $\tilde{n} \leq n$.

The bounds (30) and (33) are in contradiction for sufficiently large $\tilde{n}$, which completes the proof. ◀

E Applications, Including Bounded-Round Small-Set Disjointness

E.1 Lower Bounds

In this section we apply our new understanding of the bounded-round information complexity of EQUALITY to obtain two new lower bounds: one for OR-EQUALITY, and the other for the much-studied DISJOINTNESS problem with small-sized sets. As we shall see, both lower bounds are arguably tight.

▶ **Theorem 52** (Lower Bound for Or-Equality). *Let $k, n, r \in \mathbb{N}$ and $\delta, \varepsilon \in [0,1]$. Put $\varepsilon' = \varepsilon + k/2^n$ and $\tilde{n} = \log \frac{1-\delta}{\varepsilon'}$. For $\delta < 1 - 8(\mathrm{ilog}^{r-2}\,\tilde{n})^{-1/8}$, we have*

$$\mathrm{R}^{(r)}_{\varepsilon,\delta}(\mathrm{OREQ}_{n,k}) \geq k \cdot \mathrm{IC}^{\mu,(r)}_{\varepsilon',\delta}(\mathrm{EQ}_n) = \Omega(k(1-\delta)^3\,\mathrm{ilog}^{r-1}\,\tilde{n})\,.$$

Proof. We just need to show the first inequality and then apply Theorem 6. That inequality is proved via standard direct sum arguments for information complexity [15, 4, 5]. In fact, the old simultaneous-message lower bound for $\mathrm{OREQ}_{n,k}$ from Chakrabarti et al. [15] applies more-or-less unchanged. For completeness, we now give a self-contained proof.

Let $\mathcal{P}$ be an r-round protocol for $\mathrm{OREQ}_{n,k}$ with $\mathrm{rerr}(\mathcal{P}) \leq \varepsilon$, $\mathrm{verr}(\mathcal{P}) \leq \delta$, and $\mathrm{R}^{(r)}_{\varepsilon,\delta}(\mathrm{OREQ}_{n,k}) \geq \max\{\mathrm{rcost}(\mathcal{P}), \mathrm{vcost}(\mathcal{P})\}$. Alice and Bob solve EQ_n by the following protocol $\mathcal{Q}_j$, where j is some fixed index in $\{1, 2, \ldots, k\}$. Given an input $(x, y) \in \{0,1\}^n \times \{0,1\}^n$, they generate $\mathbf{X} := (X_1, \ldots, X_k) \sim \xi^{\otimes k}$ and $\mathbf{Y} := (Y_1, \ldots, Y_k) \sim \xi^{\otimes k}$ respectively, using private coins. They "plug in" x and y into the jth coordinates of $\mathbf{X}$ and $\mathbf{Y}$ respectively, thereby creating

$$\mathbf{Z}_{j,x} := (X_1, \ldots, X_{j-1}, x, X_{j+1}, \ldots, X_k) \text{ and } \mathbf{W}_{j,y} := (Y_1, \ldots, Y_{j-1}, y, Y_{j+1}, \ldots, Y_k)\,,$$

respectively. Finally, they emulate $\mathcal{P}$ on input $(\mathbf{Z}_{j,x}, \mathbf{W}_{j,y})$. Observe that

$$\mathrm{OREQ}_{n,k}(\mathbf{Z}_{j,x}, \mathbf{W}_{j,y}) \neq \mathrm{EQ}_n(x,y) \implies (x \neq y) \wedge \left(\exists\, i \in [k] \setminus \{j\} : X_i = Y_i\right).$$

Therefore, $\mathrm{verr}(\mathcal{Q}_j) \leq \mathrm{verr}(\mathcal{P}) \leq \delta$ and, by a union bound,

$$\mathrm{rerr}(\mathcal{Q}_j) \leq \mathrm{rerr}(\mathcal{P}) + \sum_{i=1}^{n} \Pr[X_i = Y_i] \leq \varepsilon + k/2^n = \varepsilon'\,.$$

Since $\mathcal{Q}_j$ solves EQ_n with these error guarantees, it follows that $\mathrm{icost}^{\mu}(\mathcal{Q}_j) \geq \mathrm{IC}^{\mu,(r)}_{\varepsilon',\delta}(\mathrm{EQ}_n)$.

Now, let $(X, Y) \sim \mu$ and let $\mathfrak{R}$ denote the public randomness used by $\mathcal{P}$. We can now lower bound $\mathrm{R}^{(r)}_{\varepsilon,\delta}(\textsc{oreq}_{n,k})$ as follows:

$$\begin{aligned}
\mathrm{R}^{(r)}_{\varepsilon,\delta}(\textsc{oreq}_{n,k}) &\geq \max_{x_1,\ldots,x_k,y_1,\ldots,y_k \in \{0,1\}^{kn} \times \{0,1\}^{kn}} \mathrm{cost}(\mathcal{P}; x_1, \ldots, x_k, y_1, \ldots, y_k) \\
&\geq \mathbb{E}[\mathrm{cost}(\mathcal{P}; X_1, \ldots, X_k, Y_1, \ldots, Y_k)] \\
&\geq \mathrm{H}(\mathcal{P}(X_1, \ldots, X_k, Y_1, \ldots, Y_k)) \qquad (34) \\
&\geq \mathrm{I}(\mathcal{P}(X_1, \ldots, X_k, Y_1, \ldots, Y_k) : X_1 Y_1 \ldots X_k Y_k \mid \mathfrak{R}) \\
&\geq \sum_{j=1}^{k} \mathrm{I}(\mathcal{P}(X_1, \ldots, X_k, Y_1, \ldots, Y_k) : X_i, Y_i \mid \mathfrak{R}) \qquad (35) \\
&= \sum_{j=1}^{k} \mathrm{I}(\mathcal{Q}_j(X, Y) : XY \mid \mathfrak{R}) \qquad (36) \\
&= \sum_{j=1}^{k} \mathrm{icost}^{\mu}(\mathcal{Q}_j) \geq k \cdot \mathrm{IC}^{\mu,(r)}_{\varepsilon',\delta}(\textsc{eq}_n)\,,
\end{aligned}$$

where (34) uses Fact 29 and (35) uses the independence of $\{X_1Y_1, \ldots, X_kY_k\}$ and the resulting subadditivity of mutual information, and (36) holds because, for all $j \in [k]$, the distributions of $(\mathcal{Q}_j(X, Y), X, Y, \mathfrak{R})$ and $(\mathcal{P}(X_1, \ldots, X_k, Y_1, \ldots, Y_k), X_j, Y_j, \mathfrak{R})$ are identical. This completes the proof. ◀

By plugging in $\varepsilon = 0$, $\delta = 0$ in Theorem 52 we obtain the following corollary.

▶ **Corollary 53.** $\mathrm{R}^{(r)}_{0,0}(\textsc{oreq}_{n,k}) = \Omega(k \operatorname{ilog}^{r-1}(n - \log k))$. ◀

Armed with the above lower bound, we now derive a lower bound for k-DISJ via a simple reduction, which is probably folklore. For completeness, we again give a formal proof. Note that the reduction interchanges verification and refutation errors.

▶ **Lemma 54** (Reduction from OREQ to k-DISJ)**.** *Let k, N be integers such that $N \geq k^c$ for some constant $c > 2$. Let $n = \left\lfloor \log\left(\frac{N}{k}\right) \right\rfloor$. If there exists a protocol $\mathcal{P}$ for k-$\textsc{disj}_N$ then there exists a protocol $\mathcal{Q}$ for $\textsc{oreq}_{n,k}$ such that $\mathrm{rerr}(\mathcal{Q}) \leq \mathrm{verr}(\mathcal{P})$ and $\mathrm{verr}(\mathcal{Q}) \leq \mathrm{rerr}(\mathcal{P})$ and $\mathrm{vcost}(\mathcal{Q}) \leq \mathrm{rcost}(\mathcal{P})$ and $\mathrm{rcost}(\mathcal{Q}) \leq \mathrm{vcost}(\mathcal{P})$.*

Proof. Given an input instance $(x_1, \ldots, x_k, y_1, \ldots, y_k)$ of $\textsc{oreq}_{n,k}$, we can transform it into an instance (A, B) of k-$\textsc{disj}_N$ as follows:

$$\begin{aligned}
A &= \{x_1,\ x_2 + 2^n,\ x_3 + 2 \cdot 2^n,\ \ldots,\ x_k + (k-1)2^n\} \\
B &= \{y_1,\ y_2 + 2^n,\ y_3 + 2 \cdot 2^n,\ \ldots,\ y_k + (k-1)2^n\}\,.
\end{aligned}$$

It is easy to observe that $A \cap B \neq \emptyset$ iff $\exists\, i \in [k]$ such that $x_i = y_i$ because $x_i \in \{0, 1, \ldots, 2^n - 1\}$. Therefore, $\textsc{oreq}_{n,k}(x_1, \ldots, x_k, y_1, \ldots, y_k) = \neg\, k$-$\textsc{disj}_N(A, B)$, which completes the proof. ◀

▶ **Corollary 55.** *We have $\mathrm{R}^{(r)}_{\delta,\varepsilon}(k\text{-}\textsc{disj}_N) \geq \mathrm{R}^{(r)}_{\varepsilon,\delta}(\textsc{oreq}_{\lfloor \log(N/k) \rfloor, k})$.* ◀

Combining Corollary 55 with Theorem 52, we arrive at the following theorem.

▶ **Theorem 56** (Lower Bound for k-Disjointness)**.** *Let $k, N, r \in \mathbb{N}$, $\varepsilon, \delta \in [0, 1]$ and $c > 2$ be such that $N \geq k^c$ and $\delta < 1 - 8(\operatorname{ilog}^{r-2} \tilde{n})^{-1/8}$, where $\tilde{n} = \log \frac{1-\delta}{\varepsilon + k^2/N}$. Then*

$$\mathrm{R}^{(r)}_{\delta,\varepsilon}(k\text{-}\textsc{disj}_N) = \Omega(k(1-\delta)^3 \operatorname{ilog}^{r-1} \tilde{n})\,.$$

In particular, with $\delta = 1 - \Omega(1)$ and $\varepsilon \leq k^{-\Theta(1)}$, we have $\mathrm{R}^{(r)}_{\delta,\varepsilon}(k\text{-}\textsc{disj}_N) = \Omega(k \operatorname{ilog}^r k)$. ◀

By plugging in $\varepsilon = \delta = 0$ above we arrive at a further special case that is worth highlighting.

▶ **Corollary 57.** *With $N \geq k^{2+\Omega(1)}$, we have $\mathrm{R}^{(r)}_{0,0}(k\text{-DISJ}_N) = \Omega(k \operatorname{ilog}^r k)$.* ◀

E.2 Tightness

Our lower bounds in Section E.1 have the weakness that they apply only in zero-error or small-error settings. However, they are still tight in the following sense. We can design protocols that give matching *upper* bounds under similarly small error settings. For OREQ, we give such a protocol below. For k-DISJ, a suitable analysis of a recent protocol of Sağlam and Tardos[42] gives similar results.

▶ **Theorem 58.** *For all $r < \log^* k$, there exists a r-round protocol $\mathcal{P}$ for $\mathrm{OREQ}_{n,k}$ with worst-case communication cost $O(k \operatorname{ilog}^r k)$, $\mathrm{rerr}(\mathcal{P}) < 2^{-\prod_{j=1}^{r} \operatorname{ilog}^j k}$, and $\mathrm{verr}(\mathcal{P}) = 0$.*

Proof. For ease of presentation, we give the details for a slightly weaker result, with refutation error $< k^{-10}$.

We begin with a high-level sketch of the proof, before giving formal proof details. Alice begins the protocol by sending, in parallel, k different t-bit equality tests, one for each of her inputs. Note that for any i where $x_i \neq y_i$, Bob witnesses non-equality with probability $1 - 2^{-t}$. Assuming $\mathrm{OREQ}_{n,k}(x, y) = 0$, there will be roughly $k/2^t$ coordinates i where $x_i \neq y_i$ has not yet been witnessed. Bob now tells Alice which of his coordinates remain "alive" and sends t'-bit equality tests for each of *these* coordinates, where $t' = 2^t$. Note that Bob's overall communication is roughly k bits, and that after receiving this message, Alice witnesses non-equality on all but a $2^{-t'}$-fraction of unequal pairs. In each round, players end up sending an exponentially longer equality test on an exponentially smaller number of coordinates. When communication ends, players output $\mathrm{OREQ}(x_1, \ldots, x_k, y_1, \ldots, y_k) = 1$ unless $x_i \neq y_i$ has been witnessed for all i. One potential issue with the above protocol is that too many coordinates could remain, and players wouldn't be able to communicate exponentially more bits about the remaining coordinates. This could happen both when an unusually large number of equality tests fail, or just for the simple reason that $x_i = y_i$ for many coordinates. In either case, the players simply abort and output $\mathrm{OREQ}_{n,k} = 1$. This will cause an increase in error, but the increase will be small, and it will only increase the false positive rate. A formal proof lies below.

The protocol proceeds in a number of rounds. Throughout, players maintain a vector $w \in \{0,1\}^k$ (initialized to $w = 1^k$), where $w_i = 0$ iff $x_i \neq y_i$ has been witnessed. Coordinate i is deemed "live" if $w_i = 1$.

In the first round of communication, Alice sends a $(2 \operatorname{ilog}^r k)$-bit equality test for each of the k live coordinates, at a total cost of $O(k \operatorname{ilog}^r k)$ bits.

In the jth round of communication ($1 < j < r$), the player to speak first updates her copy of w by considering the $(j-1)$th message: for for each live i, she sets $w_i = 0$ if $x_i \neq y_i$ is witnessed. Now, if more than $2k/\operatorname{ilog}^{r+1-j} k$ coordinates remain live, she sends "1", signifying that the protocol should abort and output $\mathrm{OREQ}_{n,k} = 1$. Otherwise, she sends "0", followed by her updated copy of w, followed by a $(2 \operatorname{ilog}^{r+1-j} k)$-bit equality test for each live coordinate. Thus the jth message is $O(k)$ bits long.

The final round of communication is similar, except that the equality tests are $(12 \log k)$-bits long rather than $2 \operatorname{ilog}^{r+1-r} k = 2 \log k$ bits. The receiver of the final message updates his copy of w, evaluates each equality test, and outputs $\mathrm{OREQ}_{n,k} = 1$ if any coordinates remain live. Otherwise, he outputs $\mathrm{OREQ}_{n,k} = 0$.

The overall communication is thus $O(k \operatorname{ilog}^r k)$ bits. Note also that the protocol outputs $\textsc{oreq}_{n,k} = 0$ only when $x_i \neq y_i$ was witnessed for every i. Thus, the protocol produces no false negatives.

A false positive can happen for one of two reasons: either the protocol aborts (outputting $\textsc{oreq}_{n,k} = 1$), or one or more coordinates remain live at the end of the protocol, despite having $x_i \neq y_i$ for all i.

In the former case, note that (conditioned on not aborting before round j) we have at most $2k/\operatorname{ilog}^{r+1-j} k$ live coordinates during round j. Players execute a $(2\operatorname{ilog}^{r+1-j} k)$-bit equality test during this round. Thus, a coordinate remains live after this test with probability at most $2^{-2\operatorname{ilog}^{r+1-j} k} < 1/\operatorname{ilog}^{r-j} k$. Therefore, we expect at most $k/\operatorname{ilog}^{r-j} k$ coordinates to be live in the next round. By a (crude) Chernoff bound argument, the probability of aborting during round $j+1$ (again, conditioned on not previously aborting) is less than k^{-20}, and the overall probability of aborting before the end of the protocol is less than k^{-12} (say).

In the latter case, note that the final equality test uses $12 \log k$ bits per coordinate. Therefore, players *fail* to witness $x_i \neq y_i$ with probability at most $2^{-12 \log k} = k^{-12}$. By a union bound, the overall false positive rate is at most k^{-10}. ◀

F Direct Sum for Equality with Constant Error

In this section we prove our results for PRIVATE-INTERSECTION. In the proof we will use the following modification of the strong direct sum theorem of [37] (Theorem 2.1), which uses protocols with abortion (see definitions in Appendix A.2). The simulation procedure used in the proof of this theorem in [37] preserves the number of rounds in the protocol, which allows us to state their theorem as:

▶ **Theorem 59** (Strong Direct Sum [37]). *Let $\delta \leq 1/3$. Then for every function $f\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ and distribution λ on $\mathcal{X} \times \mathcal{Y} \times \mathrm{D}$ with marginal μ_p on $\mathcal{X} \times \mathcal{Y}$ and marginal ν_p on D, such that μ_p is partitioned by ν_p, it holds that $\mathrm{IC}_{\delta}^{\mu_p^k,(r)}(f^k|\nu_p^k) \geq \Omega(k)\,\mathrm{IC}_{\frac{1}{20},\frac{1}{10},\frac{\delta}{k}}^{\mu_p,(r)}(f|\nu_p)$.*

Using the direct sum above it remains to show the following:

▶ **Lemma 60.** *There exists a distribution on $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ with marginals μ_p on $\mathcal{X} \times \mathcal{Y}$ and ν_p on $\mathcal{D}$, such that ν_p partitions μ_p and $\mathrm{IC}_{1/20,1/10,\delta/k}^{\mu_p,(r)}(\mathrm{EQ}_{n/k}|\nu_p) = \Omega(\operatorname{ilog}^r k)$.*

Proof. In the proof we can use the same hard distribution as in [37]. Let $\ell = n/k$. To construct μ_p and ν_p, let D_0 be a random variable uniformly distributed on $\{0,1\}$ and let $\mathbf{D}$ be a random variable uniformly distributed on $\{0,1\}^\ell$. Let $(\mathbf{X}, \mathbf{Y})$ be a random variable supported on $\{0,1\}^\ell \times \{0,1\}^\ell$ such that, conditioned on $D_0 = 0$ we have $\mathbf{X}$ and $\mathbf{Y}$ distributed independently and uniformly on $\{0,1\}^\ell$, and conditioned on $D_0 = 1$ we have $\mathbf{X} = \mathbf{Y} = \mathbf{D}$. Let μ_p be the distribution of $(\mathbf{X}, \mathbf{Y})$ and let ν_p be the distribution of $(D_0\mathbf{D})$. Note that ν_p partitions μ_p. Also, this distribution satisfies that $\Pr[\mathbf{X} = \mathbf{Y}] \geq 1/3$ and $\Pr[\mathbf{X} \neq \mathbf{Y}] \geq 1/3$.

Let W be a random variable distributed according to ν_p. Let E be an indicator variable over the private randomness of $\mathcal{P}$ which is equal to 1 if and only if conditioned on this private randomness $\mathcal{P}$ satisfies that it aborts with probability at most $1/10$ and succeeds with probability at least $1 - \delta/k$ conditioned on non-abortion. Given such protocol with abortion $\mathcal{P}$ we transform it into a protocol $\mathcal{P}'$ which never aborts, has almost the same information complexity and gives correct output on non-equal instances with high probability, while being correct on equal instances with constant probability. This is done by constructing $\mathcal{P}'$ so that whenever $\mathcal{P}$ outputs "abort", the output of $\mathcal{P}'$ is $X \neq Y$, otherwise $\mathcal{P} = \mathcal{P}'$. Under the distribution μ_p conditioned on the event $E = 1$ the protocol $\mathcal{P}'$ has the property that if

$X \neq Y$, then it outputs $X = Y$ with probability at most $(1/k)/\Pr_{\mu_p}[X \neq Y] \leq 3/k$. However, if $X = Y$, then the protocol may output $X \neq Y$ with probability $1/10 + (1/k)/\Pr_{\mu'_p}[X = Y] \leq 1/10 + 3/k \leq 1/5$, where the latter follows for $k \geq 30$. Thus, conditioned on $E = 1$, the protocol $\mathcal{P}'$ has failure probability $\epsilon = 1/k$ on non-equal instances $X \neq Y$, and constant failure probability $\delta = 1/5$ on equal instances $X = Y$, as desired. In this regime we can use Theorem 6. We have:

$$\begin{aligned}\mathrm{IC}^{\mu_p,(r)}_{1/20,1/10,\delta/k}(\mathrm{EQ}_{n/k}\,|\nu_p) &\geq \mathrm{I}(\mathcal{P} : X, Y|W) \\ &= \Omega(\mathrm{I}(\mathcal{P} : X, Y|W, E = 1)) - 1 \\ &= \Omega(\mathrm{I}(\mathcal{P}' : X, Y|W, E = 1)) - 2.\end{aligned}$$

Here the inequality is by definition of information compelxity and the equalities follows from Proposition 17 together with the fact that $H(E) \leq 1$, $\Pr[E = 1] = 19/20$, and the fact that the transcripts of the protocols $\mathcal{P}$ and $\mathcal{P}'$ only differ in a single bit. The right-hand side can be bounded as follows.

$$\mathrm{I}(\mathcal{P}' : X, Y|W, E = 1)) = \Omega(\mathrm{IC}^{\mu,(r)}_{1/k,1/5}(\mathrm{EQ}_{n/k})). \tag{37}$$

This follows from the construction of the distributions μ_p and ν_p that we use. If $D_0 = 0$ then $\mathbf{X} = \mathbf{Y}$ and the information revealed by $\mathcal{P}$ is equal to zero. Otherwise, if $D_0 = 1$ then the distribution of $(\mathbf{X}, \mathbf{Y})$ is uniform. Because the latter happens with probability $1/2$ we have $\mathrm{I}(\mathcal{P}' : X, Y|W, E = 1)) \geq 1/2 \cdot \mathrm{IC}^{\mu,(r)}_{1/k,1/5}(\mathrm{EQ}_{n/k}))$ as desired.

Using (37) we have $\mathrm{IC}^{\mu_p,(r)}_{1/20,1/10,\delta/k}(\mathrm{EQ}_{n/k}\,|\nu_p) = \Omega(\mathrm{IC}^{\mu,(r)}_{1/k,1/5}(\mathrm{EQ}_{n/k}))$. The proof is completed by noting that setting $\epsilon = 1/k$ and $\delta = 1/5$ in Theorem 6 gives $\mathrm{IC}^{\mu,(r)}_{1/k,1,5}(\mathrm{EQ}_{n/k}) = \Omega(\mathrm{ilog}^r\, k)$. ◀

#BIS-Hardness for 2-Spin Systems on Bipartite Bounded Degree Graphs in the Tree Non-uniqueness Region*

Jin-Yi Cai[1], Andreas Galanis[2], Leslie Ann Goldberg[2], Heng Guo[1], Mark Jerrum[3], Daniel Štefankovič[4], and Eric Vigoda[5]

1 **University of Wisconsin-Madison, Madison, WI, 53706, USA**
jyc@cs.wisc.edu, hguo@cs.wisc.edu
2 **University of Oxford, Wolfson Building, Parks Road, Oxford, OX1 3QD, UK**
andreas.galanis@cs.ox.ac.uk, leslie.goldberg@cs.ox.ac.uk
3 **Queen Mary, University of London, Mile End Road, London E1 4NS, UK**
m.jerrum@qmul.ac.uk
4 **University of Rochester, Rochester, NY, 14627, USA**
stefanko@cs.rochester.edu
5 **Georgia Institute of Technology, Atlanta, GA, 30332, USA**
vigoda@cc.gatech.edu

Abstract

Counting independent sets on bipartite graphs (#BIS) is considered a canonical counting problem of intermediate approximation complexity. It is conjectured that #BIS neither has an FPRAS nor is as hard as #Sat to approximate. We study #BIS in the general framework of two-state spin systems in bipartite graphs. Such a system is parameterized by three numbers (β, γ, λ), where β (respectively γ) represents the weight of an edge (or "interaction strength") whose end points are of the same 0 (respectively 1) spin, and λ is the weight of a 1 vertex, also known as an "external field". By convention, the edge weight with unequal 0/1 end points and the vertex weight with spin 0 are both normalized to 1. The partition function of the special case $\beta = 1$, $\gamma = 0$, and $\lambda = 1$ counts the number of independent sets. We define two notions, *nearly-independent phase-correlated spins* and *symmetry breaking*. We prove that it is #BIS-hard to approximate the partition function of any two-spin system on bipartite graphs supporting these two notions.

As a consequence, we show that #BIS on graphs of degree at most 6 is as hard to approximate as #BIS without degree bound. The degree bound 6 is the best possible as Weitz presented an FPTAS to count independent sets on graphs of maximum degree 5. This result extends to the hard-core model and to other anti-ferromagnetic two-spin models. In particular, for all antiferromagnetic two-spin systems, namely those satisfying $\beta\gamma < 1$, we prove that when the infinite $(\Delta - 1)$-ary tree lies in the non-uniqueness region then it is #BIS-hard to approximate the partition function on bipartite graphs of maximum degree Δ, except for the case $\beta = \gamma$ and $\lambda = 1$. The exceptional case is precisely the antiferromagnetic Ising model without an external field, and we show that it has an FPRAS on bipartite graphs. Our inapproximability results match the approximability results of Li et al., who presented an FPTAS for general graphs of maximum degree Δ when the parameters lie in the uniqueness region.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Spin systems, approximate counting, complexity, #BIS-hardness, phase transition

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.582

* Full version [8].

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 582–595

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

1 Introduction

There has been great progress in classifying the complexity of counting problems recently. One important success is for counting constraint satisfaction problems (#CSP), where a sweeping complexity dichotomy is proved [1, 11, 3]. While the landscape of exact counting becomes clearer, the complexity of approximate counting remains mysterious. Two typical classes of problems have been identified: 1) those that have a fully polynomial-time randomized approximation scheme (FPRAS), and 2) those that are #Sat-hard with respect to approximation preserving reductions (AP-reductions) [9]. If NP $\neq$ RP then #Sat admits no FPRAS[1] [32], and therefore neither does any problem in the second class. These two classes are analogous to P-time tractable vs. NP-hard decision or optimization problems.

Interestingly, in approximate counting, there has emerged a third distinct class of natural problems, which seems to be of intermediate complexity. It is conjectured [9] that the problems in this class do not have an FPRAS but that they are not as hard as #Sat to approximate. A canonical problem in this class has been identified, which is to count the number of independent sets in a bipartite graph (#BIS). Despite many attempts, nobody has found an FPRAS for #BIS or an AP-reduction from #Sat to #BIS. The conjecture is that neither exists. Mossel et al. [27] showed that the Gibbs sampler for sampling independent sets in bipartite graphs mixes slowly even on bipartite graphs of degree at most 6. Another interesting attempted Markov Chain for #BIS by Ge and Stefankovic [15] was also shown later to be slowly mixing by Goldberg and Jerrum [18].

#BIS plays an important role in classifying counting problems with respect to approximation. A trichotomy theorem is shown for the complexity of approximately solving unweighted Boolean counting CSPs, where in addition to problems that are solvable by FPRASes and those that are AP-reducible from #Sat, there is the intermediate class of problems which are equivalent to #BIS [10]. Many counting problems are shown to be #BIS-hard and hence are conjectured to have no FPRAS [2, 7], including estimating the partition function of the the ferromagnetic Potts model [19]. Moreover, under AP-reductions #BIS is complete in a logically defined class of problems, called #RHII$_1$, to which an increasing variety of problems have been shown to belong. Other typical complete problems in #RHII$_1$ include counting the number of downsets in a partially ordered set [9] and computing the partition function of the ferromagnetic Ising model with local external fields [17].

The problem of counting independent sets (#IS) can be viewed as a special case in the general framework of spin systems, which originated from statistical physics to model interactions between neighbors on graphs. In this paper, we focus on two-state spin systems. In general such a system is parameterized by edge weights $\beta, \gamma \geq 0$ and a vertex weight $\lambda > 0$. An instance is a graph $G = (V, E)$. A configuration σ is a mapping $\sigma : V \to \{0, 1\}$ from vertices to (two) spins. The weight $w(\sigma)$ of a configuration σ is given by

$$w(\sigma) = \beta^{m_0(\sigma)}\gamma^{m_1(\sigma)}\lambda^{n_1(\sigma)} \tag{1}$$

where $m_0(\sigma)$ is the number of $(0, 0)$ edges given by the configuration σ, $m_1(\sigma)$ is the number of $(1, 1)$ edges, and $n_1(\sigma)$ is the number of vertices assigned 1. We are interested in computing the partition function, which is defined by

$$Z_G(\beta, \gamma, \lambda) = \sum_{\sigma:V\to\{0,1\}} w(\sigma). \tag{2}$$

[1] In fact, Zuckerman proves a stronger result—there is no FPRAS for the logarithm of the number of satisfying assignments unless NP=RP.

The partition function is the normalizing factor of the Gibbs distribution, which is the distribution in which a configuration σ is drawn with probability $\Pr_{G;\beta,\gamma,\lambda}(\sigma) = \frac{w(\sigma)}{Z_G(\beta,\gamma,\lambda)}$. The spin system is called *ferromagnetic* if $\beta\gamma > 1$ and *antiferromagnetic* if $\beta\gamma < 1$. In particular, when $\beta = \gamma$, such a system is the famous *Ising model*, and when $\beta = 1$ and $\gamma = 0$, it is the *hard-core gas model*, the partition function of which counts independent sets when $\lambda = 1$. The external field λ is typically referred to as the activity or fugacity of the hard-core model.

Approximating the partition function of the hard-core model is especially well studied. We now know that the complexity transition from easy to hard corresponds exactly to the uniqueness of the Gibbs measure in infinite $(\Delta - 1)$-ary trees $\mathbb{T}_\Delta$ (for details of these notions, see [16]). Notice that $(\Delta - 1)$-ary trees are graphs of maximum degree Δ, hence our use of the notation $\mathbb{T}_\Delta$. On the algorithmic side, Weitz presented a fully polynomial-time approximation scheme (FPTAS) for the hard-core gas model on graphs of maximum degree Δ when uniqueness holds [31]. On the other hand, Sly showed that the approximation problem is #Sat-hard for a small interval beyond the uniqueness threshold [29]. Building on their work, it is now established that for all antiferromagnetic 2-spin systems there is an FPTAS for graphs of maximum degree Δ up to the uniqueness threshold [25] (see also [24, 28]), whereas non-uniqueness implies #Sat-hardness under AP-reductions on Δ-regular graphs [30] (see also [4, 13]). The only place that remains unclear is exactly at the uniqueness threshold.

A key feature of spin systems in the antiferromagnetic non-uniqueness region is the ability to support a gadget with many vertices whose spins are highly correlated with the phase of the gadget (which is either $+$ or $-$), but are nearly independent among themselves conditioned on that phase. Such a gadget was used by Sly [29] to show inapproximability of the partition function of the hard-core model when λ is just above the uniqueness threshold. A different gadget with similar properties was used by Goldberg et al. [20] to show inapproximability on a planar graph when λ is much larger. We abstract this notion of nearly-independent phase-correlated spins. It is this feature that enables us to reduce from #Sat to approximating the partition function of antiferromagnetic two-spin systems in the non-uniqueness region.

Restricted to bipartite graphs, it appears that supporting nearly-independent phase-correlated spins alone is not enough to imply #BIS-hardness. It was shown that Sly's gadget is applicable to the antiferromagnetic Ising model without an external field by Galanis et al. [13]. However, such a system has an FPRAS on bipartite graphs. The reason is that this system is perfectly symmetric on bipartite graphs and therefore can be translated into a ferromagnetic Ising system, whose partition function can be approximated using the FPRAS of Jerrum and Sinclair [22] (see Corollary 13 in the full version [8] for details). To get around this perfectly symmetric case, we introduce the second key concept called symmetry breaking. Symmetry breaking does not refer to whether the parameters of the model are symmetric, but rather whether a gadget can be constructed with a distinguished degree 1 vertex that has a certain asymmetry. Formal definitions of the two notions – nearly-independent phase-correlated spins and symmetry breaking – can be found in Section 3. Our main technical theorem is the following.

▶ **Theorem 1.** *Suppose a tuple of parameters $(\beta, \gamma, \lambda, \Delta)$ with $\beta\gamma \neq 1$ and $\Delta \geq 3$ supports nearly-independent phase-correlated spins and symmetry-breaking. Then the partition function (2) of two-spin systems (β, γ, λ) is #BIS-hard to approximate on bipartite graphs with maximum degree Δ.*

Previous hardness proofs for the problem #IS and for the problem of estimating the partition function of antiferromagnetic 2-spin systems typically reduce from Max-Cut or from the problem of counting certain types of cuts [21, 29, 30]. However such a technique

sheds little light in the bipartite setting as cut problems are trivial on bipartite graphs. Reductions between #BIS-equivalent problems typically involve transformations that "blow up" vertices and edges into sets of vertices that are completely connected, so they do not apply to bounded-degree graphs either.

A key property of Sly's gadget is that either phase occurs with probability bounded below by an inverse polynomial. This bound is sufficient in Sly's setting to reduce from Max-Cut, but it is not enough to construct AP-reductions for our use. We resolve this issue by introducing a balancing construction. The construction takes two copies of a gadget with nearly-independent phase-correlated spins, and produces a new gadget with similarly-correlated spins, but in the new gadget the two phases occur with probability close to 1/2.

The proof of Theorem 1 utilizes an intermediate problem, that is, computing the partition function of antiferromagnetic Ising systems with non-uniform external fields on bipartite graphs. A non-uniform external field means that the instance specifies a subset of vertices on which the external field acts. A 2-spin system with a non-uniform external field is very similar to a Boolean #CSP with one binary symmetric non-negative valued function (corresponding to edge weights) and one unary non-negative valued function (corresponding to vertex weights) (see, for example [5]).

Our reduction implements an external field, and this is where symmetry breaking comes into play. As discussed earlier, the partition function of Ising model without an external field has an FPRAS, so the symmetry breaking gadget seems necessary. In fact, we show that symmetry breaking holds for all 2-spin systems except for the Ising model without an external field or degenerate systems (i. e., systems satisfying $\beta\gamma = 1$). We also prove that all antiferromagnetic 2-spin systems support nearly-independent phase-correlated spins in the non-uniqueness region. Finally, applying Theorem 1 yields our main result:

▶ **Theorem 2.** *For all tuples of parameters $(\beta, \gamma, \lambda, \Delta)$ with $\Delta \geq 3$ and $\beta\gamma < 1$, except for the case $(\beta = \gamma, \lambda = 1)$, if the infinite Δ-regular tree $\mathbb{T}_\Delta$ is in the non-uniqueness region then approximating the partition function (2) on bipartite graphs with maximum degree Δ is* #BIS*-equivalent under AP-reductions.*

Let us now survey the approximability picture that this theorem helps establish. For general antiferromagnetic 2-spin models with soft constraints (i. e., $\beta\gamma > 0$), non-uniqueness holds if and only if $\sqrt{\beta\gamma} < \frac{\Delta-2}{\Delta}$ and $\lambda \in (\lambda_1, \lambda_2)$ for some critical values λ_1 and λ_2 depending on β, γ, and Δ (see [25, Lemma 5]). Hence, for all $\beta, \gamma > 0$ where $\beta\gamma < 1$, and all $\Delta \geq 3$ the following holds:

1. If $\sqrt{\beta\gamma} > \frac{\Delta-2}{\Delta}$, for all λ, there is an FPTAS to approximate the partition function for Δ-regular graphs [28, 25] (this extends to graphs of maximum degree Δ in an appropriate sense, see [25] for details).
2. If $\sqrt{\beta\gamma} < \frac{\Delta-2}{\Delta}$, then there exists $0 < \lambda_1 < \lambda_2$ so that:
   a. For all $\lambda \notin [\lambda_1, \lambda_2]$, there is an FPTAS to approximate the partition function for Δ-regular graphs [28, 25] (this again extends in an appropriate sense to graphs of maximum degree Δ [25]).
   b. For all $\lambda \in (\lambda_1, \lambda_2)$, it is #SAT-hard to approximate the partition function on Δ-regular graphs [30].
   c. For all $\lambda \in (\lambda_1, \lambda_2)$, it is #BIS-hard to approximate the partition function on bipartite graphs of maximum degree Δ (Theorem 2 in this paper).

For the particular case of the hard-core model the critical value (i. e., critical activity $\lambda_c(\Delta)$) is more easily stated. For the hard-core model (i. e., $\beta = 0$ and $\gamma = 1$) Kelly [23]

showed that non-uniqueness holds on $\mathbb{T}_\Delta$ if and only if $\lambda > \lambda_c(\Delta) := \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^\Delta}$. As a consequence we get the following corollary for the hard-core model.

▶ **Corollary 3.** *For all $\Delta \geq 3$, all $\lambda > \lambda_c(\Delta) = \frac{(\Delta-1)^{\Delta-1}}{(\Delta-2)^\Delta}$, it is #BIS-hard to approximate the partition function of the hard-core model on bipartite graphs of maximum degree Δ.*

We also get a corollary concerning the more general partition function as long as β and γ are less than 1 and the degree bound Δ is sufficiently large. For and β and γ satisfying $0 < \beta, \gamma < 1$ and any $\lambda > 0$, there exists a Δ such that (β, γ, λ) is in the non-uniqueness region of $\mathbb{T}_\Delta$ [25, Lemma 21.2]. This implies the following corollary.

▶ **Corollary 4.** *For every $0 < \beta, \gamma < 1$ and $\lambda > 0$, there exists a Δ such that the 2-spin system with parameters β, γ and with uniform or non-uniform external field λ on bipartite graphs with degree bound Δ is #BIS-equivalent under AP-reductions, except when $\beta = \gamma$ and $\lambda = 1$, in which case it has an FPRAS.*

More generally, for antiferromagnetic 2-spin systems we get the following picture for the complexity of approximating the partition function on general graphs. As usual there is a difficulty classifying the complexity of approximating the partition function at the boundary between uniqueness and non-uniqueness. To address this issue, for parameters $(\beta, \gamma, \lambda, \Delta)$, [25] define a notion of *up-to-Δ unique* which is equivalent to the parameters lying in the interior of the uniqueness region for the infinite $(d-1)$-ary tree $\mathbb{T}_d$ for all $3 \leq d \leq \Delta$ (see Definition 7 in [25]). Moreover, the parameters (β, γ, λ) satisfy *∞-strict-uniqueness* if it is up-to-∞ unique.[2] On the other side, we say the parameters (β, γ, λ) satisfy *∞-non-uniqueness* if for some $\Delta \geq 3$ the tree $\mathbb{T}_\Delta$ has non-uniqueness. The only gap between the notions of ∞-strict-uniqueness and ∞-non-uniqueness is the case when the parameters (β, γ, λ) are at the uniqueness/non-uniqueness threshold of $\mathbb{T}_\Delta$ for some Δ.

The following result detailing the complexity for general graphs is now established.

▶ **Corollary 5.** *For all tuples of parameters (β, γ, λ) with $\beta\gamma < 1$, the following holds:*

1. *If the parameters satisfy ∞-strict-uniqueness then there is a FPTAS for the partition function for all graphs [25, Theorem 2].*
2. *If the parameters satisfy ∞-non-uniqueness then:*
   a. *it is #SAT-hard to approximate the partition function on graphs [30].*
   b. *it is #BIS-hard to approximate the partition function on bipartite graphs (Theorem 2 in this paper).*

A recent paper of Liu et al. [26] shows that our Theorem 1 can also be used to analyse the complexity of *ferromagnetic* partition functions (where $\beta\gamma > 1$). In particular, it uses Theorem 1 to show #BIS-hardness for approximating the partition function for ferromagnetic 2-spin systems when $\beta \neq \gamma$ for sufficiently large external field λ. An interesting problem that remains open is to prove #BIS-hardness for the entire non-uniqueness region for ferromagnetic 2-spin systems with $\beta \neq \gamma$.

2 Approximation-Preserving Reductions and #BIS

We are interested in the complexity of approximate counting. Let Σ be a finite alphabet. We want to approximate the value of a function $f : \Sigma^* \to \mathbb{R}$. A *randomized approximation*

[2] To be precise, the notion of ∞-strict-uniqueness is called universally unique in [25, Definition 7]).

scheme is an algorithm that takes an instance $x \in \Sigma^*$ and a rational error tolerance $\varepsilon > 0$ as inputs, and outputs a rational number z such that, for every x and ε, $\Pr[e^{-\varepsilon} f(x) \leq z \leq e^{\varepsilon} f(x)] \geq \frac{3}{4}$. A *fully polynomial randomized approximation scheme* (FPRAS) is a randomized approximation scheme which runs in time bounded by a polynomial in $|x|$ and ε^{-1}. Note that the quantity $\frac{3}{4}$ can be changed to any value in the interval $(\frac{1}{2}, 1)$ or even $1 - 2^{-n^c}$ for a problem of size n without changing the set of problems that have fully polynomial randomized approximation schemes since the higher accuracy can be achieved with only polynomial delay by taking a majority vote of multiple samples.

Dyer *et al.* [9] introduced the notion of approximation-preserving reductions. Suppose f and g are two functions from Σ^* to $\mathbb{R}$. An *approximation-preserving reduction* (AP-reduction) from f to g is a randomized algorithm $\mathcal{A}$ to approximate f using an oracle for g. The algorithm $\mathcal{A}$ takes an input $(x, \varepsilon) \in \Sigma^* \times (0, 1)$, and satisfies the following three conditions: (i) every oracle call made by $\mathcal{A}$ is of the form (y, δ), where $y \in \Sigma^*$ is an instance of g, and $0 < \delta < 1$ is an error bound satisfying $\delta^{-1} \leq \text{poly}(|x|, \varepsilon^{-1})$; (ii) the algorithm $\mathcal{A}$ meets the specification for being a randomized approximation scheme for f whenever the oracle meets the specification for being a randomized approximation scheme for g; (iii) the run-time of $\mathcal{A}$ is polynomial in $|x|$ and ε^{-1}.

If an AP-reduction from f to g exists, we write $f \leq_{\text{AP}} g$, and say that f is *AP-reducible* to g. If $f \leq_{\text{AP}} g$ and $g \leq_{\text{AP}} f$, then we say that f and g are *AP-interreducible* or *AP-equivalent*, and write $f \equiv_{\text{AP}} g$. The problem #BIS is defined as follows.

Name. #BIS.
Instance. A bipartite graph B.
Output. The number of independent sets in B.

In this paper, we are interested in 2-spin systems over bounded degree bipartite graphs parametrized by a tuple (β, γ, λ). We say a real number z is *efficiently approximable* if there is an FPRAS for the problem of computing z. Throughout the paper we only deal with non-negative real parameters that are efficiently approximable. For efficiently approximable non-negative real numbers β, γ, λ and a positive integer Δ, we define the problem of computing the partition function of the 2-spin system (β, γ) with external field λ on bipartite graphs of bounded degree Δ, as follows.

Name. Bi-(M-)2-Spin$(\beta, \gamma, \lambda, \Delta)$.
Instance. A bipartite (multi)graph $B = (V, E)$ with degree bound Δ.
Output. The quantity

$$Z_B(\beta, \gamma, \lambda) = \sum_{\sigma: V \to \{0,1\}} \lambda^{\sum_{v \in V} \sigma(v)} \prod_{(v,u) \in E} \beta^{(1-\sigma(v))(1-\sigma(u))} \gamma^{\sigma(v)\sigma(u)}.$$

Notice that we also introduced a multigraph version of the same problem. It will be useful later. We drop the parameter Δ when there is no degree bound, that is, Bi-2-Spin(β, γ, λ) is the same as Bi-2-Spin$(\beta, \gamma, \lambda, \infty)$.

We also found the notion of non-uniform external field useful in the reductions. The following problems are introduced as intermediate problems. We also introduce a multigraph version, but as intermediate problems we do not need the bounded degree variant.

Name. Bi-(M-)Nonuniform-2-Spin(β, γ, λ).
Instance. A bipartite (multi)graph $B = (V, E)$ and a subset $U \subseteq V$.
Output. The quantity

$$Z_{B,U}(\beta, \gamma, \lambda) = \sum_{\sigma: V \to \{0,1\}} \lambda^{\sum_{v \in U} \sigma(v)} \prod_{(v,u) \in E} \beta^{(1-\sigma(v))(1-\sigma(u))} \gamma^{\sigma(v)\sigma(u)}.$$

3 Key Properties of the Gadget

In this section we define two key concepts: nearly-independent phase-correlated spins and symmetry breaking.

We first describe the basic setup of a certain gadget. For positive integers Δ, t and n where n is even and is at least $2t$, let T^- and T^+ be disjoint vertex sets of size t and let V^- be a size-$n/2$ superset of T^- and V^+ be a size-$n/2$ superset of T^+ which is disjoint from V^-. Let $T = T^- \cup T^+$ and $V(t,n) = V^- \cup V^+$. Let $\mathcal{G}(t,n,\Delta)$ be the set of bipartite graphs with vertex partition (V^-, V^+) in which every vertex has degree at most Δ and every vertex in T has degree at most $\Delta - 1$. We refer to the vertices in T as "terminals". Vertices in T^+ are "positive terminals" and vertices in T^- are "negative terminals".

When the gadget G is drawn from $\mathcal{G}(t,n,\Delta)$, we use the notation $T(G)$ to refer to the set of terminals. Each configuration $\sigma\colon V(t,n) \to \{0,1\}$ is assigned a unique phase $Y(\sigma) \in \{-,+\}$. Roughly in our applications of the definitions below the phase of a configuration σ is π if V^π contains more vertices with spin 1 than does $V^{-\pi}$.

We define measures Q^+ and Q^-. Fix some $0 < q^- < q^+ < 1$. For any positive integer t,

- Q^+ is the distribution on configurations $\tau\colon T \to \{0,1\}$ such that, for every $v \in T^+$, $\tau(v) = 1$ independently with probability q^+ and, for every $v \in V^-$, $\tau(v) = 1$ independently with probability q^-, and
- Q^- is the distribution on configurations $\tau\colon T \to \{0,1\}$ such that, for every $v \in T^-$, $\tau(v) = 1$ independently with probability q^+ and, for every $v \in T^+$, $\tau(v) = 1$ independently with probability q^-.

To give a rough sense for the values q^- and q^+ they will correspond to the marginal probabilities of the root of an infinite tree obtained by taking limits of finite trees with appropriate boundary conditions, see Section 7 of the full version [8] for more details.

To prove the #BIS-hardness we need a gadget where the spins of the terminals are drawn from distributions close to Q^+ or Q^- conditioned on the phase $+$ or $-$.

▶ **Definition 6.** A tuple of parameters $(\beta,\gamma,\lambda,\Delta)$ **supports nearly-independent phase-correlated spins** if there are efficiently-approximable values $0 < q^- < q^+ < 1$ such that the following is true. There are functions $n(t,\varepsilon)$, $m(t,\varepsilon)$, and $f(t,\varepsilon)$, each of which is bounded from above by a polynomial in t and ε^{-1}, and for every t and ε there is a distribution on graphs in $\mathcal{G}(t,n(t,\varepsilon),\Delta)$ such that a gadget $G = (V,E)$ with terminals T can be drawn from the distribution within $m(t,\varepsilon)$ time, and the probability that the following inequalities hold is at least $3/4$:

1. The phases are roughly balanced, i. e.,

$$\Pr_{G;\beta,\gamma,\lambda}(Y(\sigma) = +) \geq \frac{1}{f(t,\varepsilon)} \text{ and } \Pr_{G;\beta,\gamma,\lambda}(Y(\sigma) = -) \geq \frac{1}{f(t,\varepsilon)}. \tag{3}$$

2. For a configuration $\sigma\colon V \to \{0,1\}$ and any $\tau\colon T \to \{0,1\}$,

$$\left|\frac{\Pr_{G;\beta,\gamma,\lambda}(\sigma|_T = \tau \mid Y(\sigma) = +)}{Q^+(\tau)} - 1\right| \leq \varepsilon \text{ and } \left|\frac{\Pr_{G;\beta,\gamma,\lambda}(\sigma|_T = \tau \mid Y(\sigma) = -)}{Q^-(\tau)} - 1\right| \leq \varepsilon. \tag{4}$$

In fact, given a gadget with the above property, one can construct a gadget where the phases are (nearly) uniformly distributed as detailed in the following definition.

▶ **Definition 7.** We say that the tuple of parameters $(\beta, \gamma, \lambda, \Delta)$ supports **balanced** nearly-independent phase-correlated spins if Definition 6 holds with (3) replaced by:

$$\Pr_{G;\beta,\gamma,\lambda}(Y(\sigma) = +) \geq \frac{1-\varepsilon}{2} \text{ and } \Pr_{G;\beta,\gamma,\lambda}(Y(\sigma) = -) \geq \frac{1-\varepsilon}{2}, \tag{5}$$

where ε is quantified as in Definition 6.

In Section 5 of the full version [8], we prove the following lemma, which shows that for essentially all 2-spin systems, Definition 6 implies Definition 7. The lemma is proved by constructing a balanced gadget by combining two unbalanced ones.

▶ **Lemma 8.** *If the parameter tuple $(\beta, \gamma, \lambda, \Delta)$ with $\beta\gamma \neq 1$ supports nearly-independent phase-correlated spins, then it supports balanced nearly-independent phase-correlated spins.*

The main technical result for proving #BIS-hardness for 2-spin antiferromagnetic systems in the tree non-uniqueness region is the following lemma, which is proved in Section 7 of the full version [8]. The proof is rather technical, and is based on a detailed analysis of Sly's gadget [29], using ideas from [14]. Once a gadget is constructed, it can be balanced using Lemma 8.

▶ **Lemma 9.** *For all $\Delta \geq 3$, all $\beta, \gamma, \lambda > 0$ where $\beta\gamma < 1$, if the infinite Δ-regular tree $\mathbb{T}_\Delta$ is in the non-uniqueness region then the tuple of parameters $(\beta, \gamma, \lambda, \Delta)$ supports balanced nearly-independent phase-correlated spins.*

The second property of the gadget is the notion of symmetry breaking which is relatively simple.

▶ **Definition 10.** We say that a tuple of parameters $(\beta, \gamma, \lambda, \Delta)$ **supports symmetry-breaking** if there is a bipartite graph H whose vertices have degree at most Δ which has a distinguished degree-1 vertex v_H such that $\Pr_{H;\beta,\gamma,\lambda}(\sigma_{v_H} = 1) \notin \{0, \lambda/(1+\lambda), 1\}$.

We will prove in Section 6 of the full version [8] that symmetry breaking holds for all 2-spin models except in two cases (where we have tractability).

▶ **Lemma 11.** *Assume $\Delta \geq 3$. The parameters $(\beta, \gamma, \lambda, \Delta)$ support symmetry breaking unless* (i) *$\beta\gamma = 1$ or* (ii) *$\beta = \gamma$ and $\lambda = 1$.*

Having Lemma 9 and Lemma 11, Theorem 2 is a straightforward consequence of Theorem 1.

4 General Reduction

In this section we prove Theorem 1. We first show how the two notions of "nearly-independent phase-correlated spins" and "symmetry-breaking" lead to #BIS-hardness.

4.1 An Intermediate Problem

The goal of this section is to show that it is #BIS-hard to approximate the partition function of antiferromagnetic Ising models with non-uniform non-trivial external fields on bipartite graphs.

▶ **Lemma 12.** *For any $0 < \alpha < 1$, $\lambda > 0$ and $\lambda \neq 1$, #BIS $\leq_{AP}$ Bi-M-Nonuniform-2-Spin$(\alpha, \alpha, \lambda)$.*

Proof. By flipping 0 to 1 and 1 to 0 for each configuration σ, we see that Bi-M-Nonuniform-2-Spin$(\alpha, \alpha, \lambda)$ is in fact the same as Bi-M-Nonuniform-2-Spin$(\alpha, \alpha, 1/\lambda)$. Hence we may assume $\lambda < 1$.

Let $M = \left(\begin{smallmatrix} \alpha & 1 \\ 1 & \alpha \end{smallmatrix}\right)$, and $\left(\begin{smallmatrix} \rho_0 \\ \rho_1 \end{smallmatrix}\right) = M \left(\begin{smallmatrix} 1 \\ \lambda \end{smallmatrix}\right) = \left(\begin{smallmatrix} \alpha+\lambda \\ 1+\alpha\lambda \end{smallmatrix}\right)$. Note that $\alpha < 1$ and $\lambda < 1$, so $\rho_1 > \rho_0$. Let $B = (V, E)$ be an input to #BIS with $n = |V|$ and $m = |E|$. Let I_B be the number of independent sets of B. Let ε be the desired accuracy of the reduction. We will construct an instance $B' = (V', E')$ with a specified vertex subset $U \subset V'$ for Bi-M-Nonuniform-2-Spin$(\alpha, \alpha, \lambda)$ such that $\exp\left(-\frac{\varepsilon}{2}\right) I_B \le Z_{B',U}(\alpha, \alpha, \lambda)/C \le \exp\left(\frac{\varepsilon}{2}\right) I_B$, where C is a quantity that is easy to approximate. Therefore it suffices to call oracle Bi-M-Nonuniform-2-Spin$(\alpha, \alpha, \lambda)$ on B' with the specified subset U with accuracy $\frac{\varepsilon}{4}$ and approximate C within $\frac{\varepsilon}{4}$.

The construction of B' involves two positive integers t_1 and t_2. Let t_1 be the least positive integer (depending on n and ε) satisfying the first equation in (6) and let t_2 be the least positive integer depending on n, ε and t_1 satisfying the second equation in (6).

$$\alpha^{2t_1} \le \frac{\varepsilon}{6 \cdot 2^n} \quad \text{and} \quad \left(\frac{\rho_0}{\rho_1}\right)^{t_2} \le \frac{\alpha^{t_1 m} \cdot \varepsilon}{6 \cdot 2^{2t_1 m + n}}. \tag{6}$$

Note that both t_1 and t_2 are bounded from above by a polynomial in n and ε^{-1}. Given the integers t_1 and t_2, the graph B' is constructed as follows. Let $W_v = \{w_{v,j} \mid 1 \le j \le t_1 \deg(v)\}$ for each $v \in V$ where $\deg(v)$ is the degree of v in B. Let $U_{v,j} = \{u_{v,j,k} \mid 1 \le k \le t_2\}$ for any $v \in V$ and $1 \le j \le t_1 \deg(v)$. Let $W = \bigcup_{v \in V} W_v$ and $U = \bigcup_{v \in V} \bigcup_{1 \le j \le t_1 \deg(v)} U_{v,j}$. The vertex set of B' is $V' = V \cup U \cup W$. Note that $|W| = 2t_1 m$ and $|U| = 2t_1 t_2 m$.

We add t_1 parallel edges in B' between u and v for each $(u, v) \in E$ and add edges between v and every vertex in W_v, and between $w_{v,j}$ and every vertex in $U_{v,j}$ for each $v \in V$ and $1 \le j \le t_1 \deg(v)$. Formally the edge set of B' is $E' = \left(\biguplus_{1 \le i \le t_1} E\right) \cup \bigcup_{v \in V} E_v \cup \bigcup_{\substack{v \in V \\ 1 \le j \le t_1 \deg(v)}} E_{v,j}$, where $\biguplus$ denotes a disjoint union as a multiset of t_1 copies of E, $E_v = \{(v, w) | w \in W_v\}$ and $E_{v,j} = \{(w_{v,j}, u) | u \in U_{v,j}\}$ for each v and j.

Let $C = \rho_1^{2t_1 t_2 m} \alpha^{t_1 m}$ and $N = \left(\begin{smallmatrix} 1 & 1 \\ 1 & \alpha^{2t_1} \end{smallmatrix}\right)$. For each $\sigma : V \cup W \to \{0, 1\}$, let $w(\sigma)$ be the contribution to $Z_{B',U}(\alpha, \alpha, \lambda)$ of configurations that are consistent with σ. First consider configurations σ such that $\sigma(w) = 1$ for all $w \in W$. Denote by Σ the set of all such configurations on $V \cup W$. Then for $\sigma \in \Sigma$,

$$w(\sigma) = \rho_1^{t_2 |W|} \prod_{(u,v) \in E} (M_{1,\sigma(u)} M_{\sigma(u),\sigma(v)} M_{\sigma(v),1})^{t_1} = C \prod_{(u,v) \in E} N_{\sigma(u),\sigma(v)}.$$

Let $\Sigma^{ind} \subset \Sigma$ be the subset of configurations which induce an independent set on the vertices V and Z^{ind} be its contribution to $Z_{B',U}(\alpha, \alpha, \lambda)$. Let $\Sigma^{bad} = \Sigma \backslash \Sigma^{ind}$ and Z^{bad} be its contribution. If $\sigma \in \Sigma^{ind}$ then $w(\sigma) = C$. Otherwise, $w(\sigma) \le \alpha^{2t_1} C$. It implies

$$Z^{ind} = I_B \cdot C \qquad \text{and} \qquad Z^{bad} \le 2^n \alpha^{2t_1} C \le \frac{\varepsilon}{6} \cdot C, \tag{7}$$

since t_1 satisfies Eq. (6). Next consider configurations σ on $V \cup W$ such that $\sigma(w) = 0$ for at least one $w \in W$. Denote this set by Σ' and its contribution by Z^{small}. Then for $\sigma \in \Sigma'$,

$$w(\sigma) \le \left(\rho_0 \rho_1^{|W|-1}\right)^{t_2} \le \left(\frac{\rho_0}{\rho_1}\right)^{t_2} \rho_1^{t_2 |W|} = \left(\frac{\rho_0}{\rho_1}\right)^{t_2} \frac{C}{\alpha^{t_1 m}}.$$

It implies $Z^{small} \le 2^{2t_1 m + n} \left(\frac{\rho_0}{\rho_1}\right)^{t_2} \frac{C}{\alpha^{t_1 m}} \le \frac{\varepsilon}{6} \cdot C$, since $|\Sigma'| \le 2^{2t_1 m + n}$ and t_2 satisfies Eq. (6). Using this with Eq. (7) we have

$$Z_{B',U}(\alpha, \alpha, \lambda) = Z^{ind} + Z^{bad} + Z^{small} \le I_B \cdot C + \frac{\varepsilon}{6} \cdot C + \frac{\varepsilon}{6} \cdot C \le \exp\left(\frac{\varepsilon}{3}\right) I_B \cdot C,$$

and clearly $Z_{B',U}(\alpha,\alpha,\lambda) \geq I_B \cdot C$. It is also clear that C can be approximated accurate enough given FPRAS's for λ and α. This finishes our proof. ◀

4.2 Simulating the Antiferromagnetic Ising Model

In this section we prove the following lemma.

▶ **Lemma 13.** *Suppose β, γ and λ are efficiently approximable reals satisfying $\beta,\gamma \geq 0$, $\lambda > 0$ and $\beta\gamma \neq 1$. Suppose that Δ is either an integer that is at least 3 or $\Delta = \infty$ (indicating that we do not have a degree bound). If $(\beta,\gamma,\lambda,\Delta)$ supports nearly-independent phase-correlated spins and symmetry breaking, then there exist efficiently approximable $0 < \alpha < 1$ and $\lambda' > 0$ such that $\lambda' \neq 1$ and* Bi-M-Nonuniform-2-Spin$(\alpha,\alpha,\lambda') \leq_{AP}$ Bi-2-Spin$(\beta,\gamma,\lambda,\Delta)$.

Proof. We prove the antiferromagnetic case first, that is, $\beta\gamma < 1$. α and λ' are chosen as follows. Recall that $M = \left(\begin{smallmatrix}\beta & 1\\ 1 & \gamma\end{smallmatrix}\right)$ and $M^+ = \left(\begin{smallmatrix}1-q^- & q^-\\ 1-q^+ & q^+\end{smallmatrix}\right)$. Let $N = M^+M(M^+)^{\mathsf{T}} = \left(\begin{smallmatrix}N_{--} & N_{-+}\\ N_{+-} & N_{++}\end{smallmatrix}\right)$. Then $\det(N) = (\beta\gamma-1)(q^+-q^-)^2 < 0$. Therefore $N_{--}N_{++} < N_{-+}N_{+-}$ and let $\alpha = \frac{N_{--}N_{++}}{N_{-+}N_{+-}} < 1$. Moreover, suppose H is the symmetry breaking gadget with distinguished vertex v_H. Let $\rho = \left(\begin{smallmatrix}\rho_0\\ \rho_1\end{smallmatrix}\right)$ where ρ_i denote $\Pr_{H;\beta,\gamma,\lambda}(\sigma_{v_H} = i)$ for spin $i \in \{0,1\}$ and $\rho_0 + \rho_1 = 1$. Let $\rho' = \left(\begin{smallmatrix}\rho'_0\\ \rho'_1\end{smallmatrix}\right) = M^+\left(\begin{smallmatrix}\rho_0\\ \rho_1/\lambda\end{smallmatrix}\right)$, and $\lambda' = \frac{\rho'_1}{\rho'_0}$. It is easy to verify that $\lambda' \neq 1$ as $\rho_1 \neq \lambda/(1+\lambda)$ by the symmetry breaking assumption.

Given $0 < \varepsilon < 1$ and a bipartite multigraph $B = (V,E)$ with a subset $U \subseteq V$ where $|V| = n$, $|E| = m$, and $|U| = n'$, our reduction first constructs a bipartite graph B' with degree at most Δ. The construction of B' involves a gadget G. Since $(\beta,\gamma,\lambda,\Delta)$ supports nearly-independent phase-correlated spins, by Lemma 8 $(\beta,\gamma,\lambda,\Delta)$ also supports balanced nearly-independent phase-correlated spins. Therefore we draw $G \sim \mathcal{G}(t, n(t,\varepsilon'), \Delta)$ such that Eq. (5) and Eq. (4) hold with probability at least 3/4, where $t = m+1$ and $\varepsilon' = \frac{\varepsilon}{8n}$. Assume G satisfies them and otherwise the reduction fails. We will construct B' such that

$$\exp\left(-\frac{\varepsilon}{2}\right) Z_{B,U}(\alpha,\alpha,\lambda') \leq \frac{Z_{B'}}{(N_{+-}N_{-+})^m\,(\rho'_0 Z_H)^{n'}\left(\frac{Z_G}{2}\right)^n} \leq \exp\left(\frac{\varepsilon}{2}\right) Z_{B,U}(\alpha,\alpha,\lambda'),$$

where we use the abbreviated expressions $Z_{B'} = Z_{B'}(\beta,\gamma,\lambda)$, $Z_H = Z_H(\beta,\gamma,\lambda)$, and $Z_G = Z_G(\beta,\gamma,\lambda)$. The lemma follows by one oracle call for $Z_{B'}$ with accuracy $\frac{\varepsilon}{6}$, one oracle call for Z_G with accuracy $\frac{\varepsilon}{6n}$, and an approximation of other terms in the denominator with accuracy $\frac{\varepsilon}{6}$ using FPRASes for q^-, q^+, β, γ and λ.

The graph B' is constructed as follows. For each vertex $v \in V$ we introduce a copy of G, denoted by G_v with vertex set $V(G_v)$. Moreover, for each vertex $u \in U$ we introduce a copy of H, denoted by H_u. Whenever a terminal vertex is used in the construction once, we say it is occupied. For each $(u,v) \in E$, we connect one currently unoccupied positive (and respectively negative) terminal of G_u to one currently unoccupied positive (and respectively negative) terminal of G_v. Denote by E' all these edges between terminals. For each $u \in U$, we identify an unoccupied positive terminal of G_u with the distinguished vertex v_{H_u} of H_u. We denote this terminal by t_u. The resulting graph is B'. It is clear that B' is bipartite and has bounded degree Δ.

Let $\tilde{\sigma}\colon V \to \{-,+\}$ be a configuration of the phases of the G_v's. Let $Z_{B'}(\tilde{\sigma})$ be the contribution to $Z_{B'}$ from the configurations σ that are consistent with $\tilde{\sigma}$ in the sense that, for each $v \in V$, $Y(\sigma_{V(G_v)}) = \tilde{\sigma}(v)$. Then $Z_{B'} = \sum_{\tilde{\sigma}} Z_{B'}(\tilde{\sigma})$. Let T be the set of all terminals $T = \cup_{v\in V} T(G_v)$ and $\tau\colon T \to \{0,1\}$ be a configuration on T. Let $\tau_{T(G_v)}$ be the configuration τ restricted to $T(G_v)$. Recall that for $\pi \in \{-,+\}$, $Z^{\pi}_{G_v}(\tau_{T(G_v)})$ is the

contribution to Z_{G_v} from configurations that have phase π and are consistent with $\tau_{T(G_v)}$. Also, $\Pr_{G_v;\beta,\gamma,\lambda}(\tau_{T(G_v)} \mid Y(\sigma_{V(G_v)}) = \pi) = Z^{\pi}_{G_v}(\tau_{T(G_v)})/Z^{\pi}_{G_v}$. Moreover, for each $u \in U$ and each spin $i \in \{0,1\}$, let $Z_{H_u}(i)$ be the contribution to Z_{H_u} from configurations σ with $\sigma(t_u) = i$. Hence, $\rho_i = \Pr_{H_u;\beta,\gamma,\lambda}(\sigma(t_u) = i) = \frac{Z_{H_u}(i)}{Z_{H_u}}$. We express $Z_{B'}(\tilde{\sigma})$ as $Z_{B'}(\tilde{\sigma}) = \sum_{\tau:\, T\to\{0,1\}} w_{E'}(\tau) \prod_{v\in V} Z^{\tilde{\sigma}(v)}_{G_v}(\tau_{T(G_v)}) \prod_{u\in U} \left(Z_{H_u}(\tau(t_u))/\lambda^{\tau(t_u)}\right)$, where $w_{E'}(\tau)$ is the contribution of edges in E' given configuration τ. Notice that we divide the last factor by λ when $\tau(t_u) = 1$ because we counted the vertex weight twice in that case. Define $\widetilde{Z_{B'}}(\tilde{\sigma})$ to be an approximation version of the partition function where on each $T(G_v)$ the spins are chosen exactly according to $Q^{\tilde{\sigma}(v)}$. That is,

$$\begin{aligned}\widetilde{Z_{B'}}(\tilde{\sigma}) &= \sum_{\tau:\, T\to\{0,1\}} w_{E'}(\tau) \prod_{v\in V} Z^{\tilde{\sigma}(v)}_{G_v} Q^{\tilde{\sigma}(v)}(\tau_{T(G_v)}) \prod_{u\in U} \frac{Z_{H_u}(\tau(t_u))}{\lambda^{\tau(t_u)}} \\ &= \left(\prod_{v\in V} Z^{\tilde{\sigma}(v)}_{G_v}\right) \cdot \left(\sum_{\tau:\, T\to\{0,1\}} w_{E'}(\tau) \prod_{v\in V} Q^{\tilde{\sigma}(v)}(\tau_{T(G_v)}) \prod_{u\in U} \frac{Z_{H_u}(\tau(t_u))}{\lambda^{\tau(t_u)}}\right). \qquad (8)\end{aligned}$$

Let $\widetilde{Z_{B'}} = \sum_{\tilde{\sigma}} \widetilde{Z_{B'}}(\tilde{\sigma})$. Eq. (4) implies that $Z_{B'}(\tilde{\sigma})$ and $\widetilde{Z_{B'}}(\tilde{\sigma})$ are close, that is,

$$(1-\varepsilon')^n \le \frac{Z_{B'}(\tilde{\sigma})}{\widetilde{Z_{B'}}(\tilde{\sigma})} \le (1+\varepsilon')^n. \qquad (9)$$

Moreover Eq. (5) implies that

$$\left(\frac{1-\varepsilon'}{2}\right)^n \le \frac{\prod_{v\in V} Z^{\tilde{\sigma}(v)}_{G_v}}{(Z_G)^n} \le \left(\frac{1+\varepsilon'}{2}\right)^n. \qquad (10)$$

Notice that here Z_{G_v} is the same for any $v \in V$ as the G_v's are identical copies of G. Now, given $\tilde{\sigma}$, we calculate $\sum_{\tau:\, T\to\{0,1\}} w_{E'}(\tau) \prod_{v\in V} Q^{\tilde{\sigma}(v)}(\tau_{T(G_v)}) \prod_{u\in U} Z_{H_u}(\tau(t_u))/\lambda^{\tau(t_u)}$. As the measure $Q^{\tilde{\sigma}(v)}$ is i.i.d., we may count the weight of each edge in E' independently. Notice that $N_{\pi_1\pi_2}$ is the edge contribution when one end point is chosen with probability q^{π_1} and the other q^{π_2}. For an edge $(u,v) \in V$, if u and v are assigned the same phase $+$, then an edge in E' connecting one $+$ terminal of G_u and one $+$ terminal of G_v gives a weight of N_{++} and an edge connecting two $-$ terminals gives N_{--}. The total weight is $\mu_1 = N_{++}N_{--}$. Similarly if u and v are assigned the same phase $-$, the total weight is μ_1 as well. On the other hand if u and v are assigned distinct phases $+$ and $-$, the total weight is $\mu_2 = N_{+-}N_{-+}$. Recall that $\alpha = \frac{\mu_1}{\mu_2}$. Moreover, for each $u \in U$, if $\tilde{\sigma}(u) = +$, then the contribution of H_u is $\rho'_1 Z_{H_u}$ and otherwise $\rho'_0 Z_{H_u}$. Notice that here Z_{H_u} is the same for any $u \in U$ as the H_u's are identical copies of H. Recall that $\lambda' = \frac{\rho'_1}{\rho'_0}$. Plugging these calculations into Eq. (8), we have

$$\begin{aligned}\widetilde{Z_{B'}}(\tilde{\sigma}) &= \left(\prod_{v\in V} Z^{\tilde{\sigma}(v)}_{G_v}\right) \cdot \left(\mu_1^{m_+(\tilde{\sigma})} \mu_2^{m-m_+(\tilde{\sigma})} (\rho'_1 Z_H)^{n_+(\tilde{\sigma})} (\rho'_0 Z_H)^{n'-n_+(\tilde{\sigma})}\right) \\ &= \mu_2^m (\rho'_0 Z_H)^{n'} \left(\prod_{v\in V} Z^{\tilde{\sigma}(v)}_{G_v}\right) \cdot \left(\alpha^{m_+(\tilde{\sigma})} (\lambda')^{n_+(\tilde{\sigma})}\right), \qquad (11)\end{aligned}$$

where $m_+(\tilde{\sigma})$ denotes the number of edges of which the two endpoints are of the same phase given $\tilde{\sigma}$, and $n_+(\tilde{\sigma})$ denotes the number of vertices in U that are assigned $+$ given $\tilde{\sigma}$. Apply Eq.(10) to Eq.(11),

$$(1-\varepsilon')^n \left(\alpha^{m_+(\tilde{\sigma})} (\lambda')^{n_+(\tilde{\sigma})}\right) \le \frac{\widetilde{Z_{B'}}(\tilde{\sigma})}{\mu_2^m (\rho'_0 Z_H)^{n'} \left(\frac{Z_G}{2}\right)^n} \le (1+\varepsilon')^n \left(\alpha^{m_+(\tilde{\sigma})} (\lambda')^{n_+(\tilde{\sigma})}\right). \qquad (12)$$

Then we sum over $\tilde{\sigma}$ in Eq. (12),

$$(1-\varepsilon')^n \left(\sum_{\tilde{\sigma}} \alpha^{m_+(\tilde{\sigma})} (\lambda')^{n_+(\tilde{\sigma})} \right) \leq \frac{\widetilde{Z_{B'}}}{\mu_2^m (\rho_0' Z_H)^{n'} \left(\frac{Z_G}{2}\right)^n} \leq (1+\varepsilon')^n \left(\sum_{\tilde{\sigma}} \alpha^{m_+(\tilde{\sigma})} (\lambda')^{n_+(\tilde{\sigma})} \right). \quad (13)$$

However notice that $Z_{B,U}(\alpha, \alpha, \lambda') = \sum_{\tilde{\sigma}} \alpha^{m_+(\tilde{\sigma})} (\lambda')^{n_+(\tilde{\sigma})}$ by just mapping $-$ to 0 and $+$ to 1 in each configuration $\tilde{\sigma}$. Combine Eq.(9), and Eq.(13),

$$(1-\varepsilon')^{2n} Z_{B,U}(\alpha, \alpha, \lambda') \leq \frac{Z_{B'}}{\mu_2^m (\rho_0' Z_H)^{n'} \left(\frac{Z_G}{2}\right)^n} \leq (1+\varepsilon')^{2n} Z_{B,U}(\alpha, \alpha, \lambda').$$

Recall that $\varepsilon' = \frac{\varepsilon}{8n}$ and we get the desired bounds.

The other case is ferromagnetic, that is, $\beta\gamma > 1$. Notice that in this case $\det(N) = (\beta\gamma - 1)(q^+ - q^-)^2 > 0$, So we choose $\alpha = \frac{N_{+-}N_{-+}}{N_{++}N_{--}} < 1$ and λ' to be the same as the antiferromagnetic case. The construction of B' is similar to the previous case, with the following change. For each $(u, v) \in E$, we connect one unoccupied positive terminal of G_u to one unoccupied negative terminal of G_v, and vice versa. The rest of the construction is the same. With this change, given a configuration $\tilde{\sigma} \colon V \to \{-, +\}$, if two endpoints are assigned the same spin, the contribution is $N_{+-}N_{-+}$ and otherwise $N_{++}N_{--}$. Therefore the effective edge weight is $\alpha < 1$ when the spins are the same, after normalizing the weight to 1 when the spins are distinct. The rest of the proof is the same. ◀

4.3 Completing the Proof of Theorem 1

Proof of Theorem 1. #BIS-hardness in Theorem 1 follows directly from Lemma 12 and Lemma 13. The other direction, #BIS-easiness, follows fairly directly from Theorem 47 of [6] (the full version of [7]). An edge in the instance graph can be viewed as a constraint of arity 2. If $\beta\gamma > 1$, then the constraint on the edge is "weakly log-supermodular" and the vertex weight can be viewed as a unary constraint, which is taken as given in a "conservative" CSP. If $\beta\gamma \leq 1$, then reverse the interpretation of 0 and 1 on one side of the bipartition of the instance graph, so that the effective interaction along an edge is given by the matrix $\left(\begin{smallmatrix} 1 & \beta \\ \gamma & 1 \end{smallmatrix}\right)$. This constraint is also "weakly log-supermodular" since $1 \cdot 1 \geq \beta\gamma$. After the reversing there are two vertex weights λ and λ^{-1}, which are also allowed for "conservative" CSP instances. ◀

Acknowledgement. Jin-Yi Cai and Heng Guo are supported by NSF grant CCF-1217549. Heng Guo is also supported by a 2013 Simons award for graduate students in theoretical computer science. Andreas Galanis and Eric Vigoda are supported by NSF grant CCF-1217458. Daniel Štefankovič is supported by NSF grant CCF-1318374.

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.

References

1. Andrei A. Bulatov. The complexity of the counting constraint satisfaction problem. In *ICALP*, pages 646–661. Springer-Verlag, 2008.
2. Andrei A. Bulatov, Martin Dyer, Leslie Ann Goldberg, Mark Jerrum, and Colin Mcquillan. The expressibility of functions on the Boolean domain, with applications to counting CSPs. *J. ACM*, 60(5):32:1–32:36, October 2013.
3. Jin-Yi Cai and Xi Chen. Complexity of counting CSP with complex weights. In *STOC*, pages 909–920. ACM, 2012.
4. Jin-Yi Cai, Xi Chen, Heng Guo, and Pinyan Lu. Inapproximability after uniqueness phase transition in two-spin systems. In *COCOA*, pages 336–347, 2012.
5. Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Holant problems and counting CSP. In *STOC*, pages 715–724. ACM, 2009.
6. Xi Chen, Martin E. Dyer, Leslie Ann Goldberg, Mark Jerrum, Pinyan Lu, Colin McQuillan, and David Richerby. The complexity of approximating conservative counting CSPs. Preprint, 2012. Available from the arXiv at: http://arxiv.org/abs/1208.1783
7. Xi Chen, Martin E. Dyer, Leslie Ann Goldberg, Mark Jerrum, Pinyan Lu, Colin McQuillan, and David Richerby. The complexity of approximating conservative counting CSPs. In *STACS*, pages 148–159, 2013.
8. Jin-Yi Cai, Andreas Galanis, Leslie Ann Goldberg, Heng Guo, Mark Jerrum, Daniel Štefankovič, and Eric Vigoda. #BIS-Hardness for 2-Spin Systems on Bipartite Bounded Degree Graphs in the Tree Non-uniqueness Region. Preprint, 2014. Available from the arXiv at: http://arxiv.org/abs/1311.4451
9. Martin E. Dyer, Leslie Ann Goldberg, Catherine S. Greenhill, and Mark Jerrum. The relative complexity of approximate counting problems. *Algorithmica*, 38(3):471–500, 2003.
10. Martin E. Dyer, Leslie Ann Goldberg, and Mark Jerrum. An approximation trichotomy for Boolean #CSP. *J. Comput. Syst. Sci.*, 76(3-4):267–277, 2010.
11. Martin E. Dyer and David Richerby. An effective dichotomy for the counting constraint satisfaction problem. *SIAM J. Comput.*, 42(3):1245–1274, 2013.
12. Andreas Galanis, Qi Ge, Daniel Štefankovič, Eric Vigoda, and Linji Yang. Improved inapproximability results for counting independent sets in the hard-core model. In *APPROX-RANDOM*, pages 567–578, 2011.
13. Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Inapproximability of the partition function for the antiferromagnetic Ising and hard-core models. Preprint, 2012. Available from the arXiv at: http://arxiv.org/abs/1203.2226v3
14. Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Inapproximability for antiferromagnetic spin systems in the tree non-uniqueness region. Preprint, 2013. To appear in *STOC*, 2014. Available from the arXiv at: http://arxiv.org/abs/1305.2902v2
15. Qi Ge and Daniel Štefankovič. A graph polynomial for independent sets of bipartite graphs. In *FSTTCS*, pages 240–250, 2010.
16. Hans-Otto Georgii. *Gibbs Measures and Phase Transitions, 2nd edition*. Walter de Gruyter, 2011.
17. Leslie Ann Goldberg and Mark Jerrum. The complexity of ferromagnetic Ising with local fields. *Combinatorics, Probability & Computing*, 16(1):43–61, 2007.
18. Leslie Goldberg and Mark Jerrum. A counterexample to rapid mixing of the Ge-Stefankovic process. *Electron. Commun. Probab.*, 17:no. 5, 1–6, 2012.
19. Leslie Ann Goldberg and Mark Jerrum. Approximating the partition function of the ferromagnetic Potts model. *J. ACM*, 59(5):25, 2012.
20. Leslie Ann Goldberg, Mark Jerrum, and Colin McQuillan. Approximating the partition function of planar two-state spin systems. Preprint, 2012. Available from the arXiv at: http://arxiv.org/abs/1208.4987

21 Leslie Ann Goldberg, Mark Jerrum, and Mike Paterson. The computational complexity of two-state spin systems. *Random Struct. Algorithms*, 23(2):133–154, 2003.

22 Mark Jerrum and Alistair Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM J. Comput.*, 22(5):1087–1116, 1993.

23 Frank Kelly. Stochastic models of computer communication systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 47(3):pp. 379–395, 1985.

24 Liang Li, Pinyan Lu, and Yitong Yin. Approximate counting via correlation decay in spin systems. In *SODA*, pages 922–940, 2012.

25 Liang Li, Pinyan Lu, and Yitong Yin. Correlation decay up to uniqueness in spin systems. In *SODA*, pages 67–84, 2013. Available from the arXiv at: http://arxiv.org/pdf/1111.7064v2.pdf

26 Jingcheng Liu, Pinyan Lu, and Chihao Zhang. The complexity of ferromagnetic two-spin systems with external fields. Preprint, 2014. Available from the arXiv at: http://arxiv.org/abs/1402.4346

27 Elchanan Mossel, Dror Weitz, and Nicholas Wormald. On the hardness of sampling independent sets beyond the tree threshold. *Prob. Theory Related. Fields*, 143:401–439, 2009.

28 Alistair Sinclair, Piyush Srivastava, and Marc Thurley. Approximation algorithms for two-state anti-ferromagnetic spin systems on bounded degree graphs. In *SODA*, pages 941–953, 2012.

29 Allan Sly. Computational transition at the uniqueness threshold. In *FOCS*, pages 287–296, 2010.

30 Allan Sly and Nike Sun. The computational hardness of counting in two-spin models on d-regular graphs. In *FOCS*, pages 361–369, 2012.

31 Dror Weitz. Counting independent sets up to the tree threshold. In *STOC*, pages 140–149, 2006.

32 David Zuckerman. On unapproximable versions of NP-complete problems. *SIAM J. Comput.*, 25(6):1293–1304, 1996.

The Power of Super-logarithmic Number of Players*

Arkadev Chattopadhyay[1] and Michael E. Saks[2]

1 **School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India**
arkadev.c@tifr.res.in

2 **Department of Mathematics, Rutgers University, Piscataway, NJ, USA**
msaks30@gmail.com

Abstract

In the 'Number-on-Forehead' (NOF) model of multiparty communication, the input is a $k \times m$ boolean matrix A (where k is the number of players) and Player i sees all bits except those in the i-th row, and the players communicate by broadcast in order to evaluate a specified function f at A. We discover new computational power when k exceeds $\log m$. We give a protocol with communication cost poly-logarithmic in m, for block composed functions with limited block width. These are functions of the form $f \circ g$ where f is a symmetric b-variate function, and g is a kr-variate function and $(f \circ g)(A)$ is defined, for a $k \times br$ matrix to be $f(g(A^1), \ldots, g(A^b))$ where A^i is the i-th $k \times r$ block of A. Our protocol works provided that $k > 1 + \ln b + 2^r$. Ada et al. [2] previously obtained *simultaneous* and deterministic efficient protocols for composed functions of block-width $r = 1$. The new protocol is the first to work for block composed functions with $r > 1$. Moreover, it is simultaneous, with vanishingly small error probability, if public coin randomness is allowed. The deterministic and zero-error version barely uses interaction.

1998 ACM Subject Classification F. Theory of Computation

Keywords and phrases Communication complexity, Number-On-Forehead model, composed functions

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.596

1 Introduction

In the *Number-on-Forehead* (NOF) model of communication, k players collaborate to evaluate a function f on a $k \times m$ boolean matrix $X = (x_{i,j})$. Player i knows all input bits except those in row i which is represented metaphorically by saying that row i is on the forehead of Player i, who sees all foreheads except her own. The players communicate by broadcast. The goal is to design a communication protocol for evaluating f that minimizes the number of bits of communication. Every such function can be evaluated with $m + 1$ bits of communication by having the kth player broadcast the first row of the matrix; the first player (who then knows the entire matrix) evaluates the function and announces the result.

Since it was introduced by Chandra, Furst and Lipton [8], the model has been studied extensively (e. g., [5, 12, 3, 6, 10, 18, 9, 14, 11, 16]), in part because it captures a communication bottleneck relevant to several models of computation such as branching programs, boolean circuits, SAT refutation via polynomial calculus etc. For each of these models, proving lower

* The first author is partially supported by a Ramanujan Fellowship of the DST and research grants of DAE. The second author's work was supported in part by NSF Grants CCF-0832787 and CCF-1218711.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 596–603

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

bounds for computing some function f reduces to proving communication lower bounds for a function related to f in the NOF model.

For example, the complexity class ACC^0 is believed to be rather weak.[1] This belief is based on the famous Razborov-Smolensky Theorem [15, 17] stating that AC^0 circuits augmented with MOD_p gates, for any fixed prime p, cannot even compute efficiently the majority function MAJ (which outputs 1 if at least half the input bits are 1). A widely held conjecture says that ACC^0 does not contain MAJ, but the only known non-trivial separation is $NEXP \not\subset \mathrm{ACC}^0$ [19]. Combining results of [13, 7] gives that for any $f : \{0,1\}^n \to \{0,1\}$ in ACC^0 there is a constant C such that for $k = (\log n)^C$ if the variables of f are arranged (arbitrarily) in a matrix with k rows (padding rows with dummy inputs as needed) there is a k-player NOF protocol for evaluating f that is efficient (uses $\log(n)^{O(1)}$ bits of communication). This has inspired researchers to seek an explicit function f on $n = mk$ bits for which there is provably no efficient NOF k-party protocol as long as $k = (\log n)^{O(1)}$. This would separate ACC^0 from any complexity class containing f.

For the generalized inner product function GIP_k^m which outputs 1 if the input matrix has an odd number of all 1 columns, [5] proves a bound of $\Omega(m/4^k)$. This lower bound is $m^{\Omega(1)}$ if the number of players is less than $(1-\varepsilon)\log m$ but becomes trivial if $k \geq \log m$. Similarly all known NOF lower bounds become trivial for $k \geq \log m$.

One might guess that GIP remains hard for NOF when $k \geq \log(m)$, but surprisingly Grolmusz [12] found a protocol for $k \geq \log(m)$ with cost $O((\log m)^2)$. His protocol relies on the structure of GIP. For a k-variate boolean function g and an m variate function f the composition $\big(f \circ g\big)(M)$ has output $f(g(A^1), \ldots, g(A^m))$ where A^i is the i-th column of A. We call f the *outer function* and g the *inner function*. For GIP, f is sum modulo 2, and g is AND. In fact, Grolmusz's protocol works if f is symmetric (invariant under any permutation of the variables).

Babai et al. [4] suggested that the composed function with outer function MAJ_m (the m bit majority function) and inner function MAJ_k might be hard for NOF, however Babai, Gál, Kimmel and Lokam [3] refuted this by giving an efficient simultaneous protocol [2] that works for a composed function with symmetric outer function and an inner function that is both symmetric and *compressible*, provided that the number of players is a sufficiently large poly-logarithmic function of m. We won't define compressible here, but we note that MAJ is compressible and so their protocol applies to $\mathrm{MAJ}_m \circ \mathrm{MAJ}_k$.

Babai et al. [3] then suggested that $\mathrm{MAJ}_m \circ Q$ where Q is not compressible, might be hard for the NOF model. Recently, however, Ada et al.[2] showed that for k slightly larger than $\log m$, the composition of any symmetric function with *any* k-variate inner function, has a very efficient deterministic simultaneous NOF protocol.

Babai et al. also suggested considering composed functions whose inner function depends on more than 1 bit from each player. More precisely, let b and r be integers and let $m = br$. Split the $k \times m$ matrix A into b blocks, $A^1, \ldots, A^b$, where each block is a $k \times r$ matrix. Consider the composition $f \circ g$ where f has b variables and g has kr variables. We call r the *block width* of g. Specifically, they suggested looking at the function $\mathrm{MAJ}_b \circ T_t^{k,r}$, where $T_t^{k,r}$ takes as input a $k \times r$ matrix and interprets each row i as an r-bit integer z_i, and outputs 1 if $z_1 + \cdots + z_k > t$. They suggested $b = r$ as a case of special interest, but noted that even the case $r = 2$ is open.

[1] ACC^0 is the class of boolean functions computable by circuits of polynomial size and constant depth using AND gates, OR gates and MOD_w gates for some fixed positive integer w. A MOD_w gate outputs 1 iff the sum of the input values is divisible by w.

[2] In a simultaneous protocol, all processors simultaneously send one message to a *referee* who computes $f(A)$ from the messages.

Here we give the first efficient NOF protocol for composed functions having block width above 1. Corollary 2 implies that MAJ $\circ T_t^{k,r}$ has an efficient NOF protocol of only poly-logarithmic (i. e. $\log(m)^{O(1)}$) cost, when the number of players k is $\Omega(log(m))^2$ and the block width r is at most $\log\log(m)$. While our primary interest is in boolean functions, our result is naturally stated for polynomial functions over a finite field. The set up we work with is:

- $\mathbb{F}$ is a finite field.
- $D \subseteq \mathbb{F}$.
- $p^1, \ldots, p^b$ are polynomial functions of the entries of a $k \times m$ matrix each of which depends on at most r variables per row.
- $p = \sum_{i=1}^{b} p^i$.
- A is an assignment to the variables whose entries are all in D.
- $n = mk$.

We consider the k-party NOF complexity of evaluating $p(A)$. A key observation that was used previously in making the connection between ACC^0 lower bounds and NOF-complexity, is that if p is a polynomial of degree strictly less than k, then p has a very efficient k-party simultaneous protocol: for any monomial of degree less than k there is some player who sees all the variables of that monomial and so the polynomial p can be decomposed as a sum of polynomials $p^1 + \cdots + p^k$ where Player j sees all of the variables needed to evaluate p^j, and so can simply announce $p^j(A)$. However, if the degree of p exceeds k there are no general methods known. In the above set up, the degree of p is rk. Our main result shows that if r is not too big then we can get efficient protocols.

▶ **Theorem 1.** **1.** *Let $\gamma > 0$ and suppose $k \geq 1 + |D|^r \ln(b/\gamma)$. There is a randomized simultaneous message NOF protocol that outputs either $p(A)$ or "failure", where the probability that it outputs "failure" is at most γ. The total communication cost of the protocol is at most $\big(1 + |D|^r \ln(b/\gamma)\big)\lceil \log(1 + |\mathbb{F}|)\rceil$.*

2. *Suppose $k \geq (1 + |D|^r \ln(2b))$. There is a 2 round deterministic NOF protocol that outputs $p(A)$ having total communication cost $(1 + |D|^r \ln(2b))(\lceil r \log |D|\rceil + \lceil \log |\mathbb{F}|\rceil)$.*

▶ Remark. As in the work of Babai et al. [3], in public-coin simultaneous message protocols, all coin-tosses are visible to all players and the referee.

For boolean functions we get:

▶ **Corollary 2.** *Let g be a boolean function whose variable set is a $k \times r$ matrix and let f be a symmetric b-variate boolean function.*

- *Suppose $\gamma > 0$ and $k \geq 1 + 2^r \ln(b/\gamma)$. There is a public-coin randomized simultaneous message protocol that outputs either $(f \circ g)(A)$ or "failure", where the probability that it outputs failure is at most γ. The total communication is at most $\big(1 + 2^r \ln(b/\gamma)\big)\lceil \log(1 + 2b)\rceil$.*
- *If $k \geq 1 + 2^r \ln(2b)$, there is a 2 round deterministic NOF protocol for $f \circ g$ with communication $(1 + 2^r \ln(2b))(r + \lceil \log(2b)\rceil)$.*

To deduce the corollary, let q be the smallest prime that is greater than b (so $b \leq q \leq 2b$) and let $\mathbb{F}$ be the field of integers mod q. For any boolean function there is a polynomial λ over field $\mathbb{F}$ that agrees with g on every 0-1 input. Let λ be the kr-variate polynomial over $\mathbb{F}$ that represents the given boolean function g. Let X be a $k \times rb$ matrix of variables. For $i \in [b]$, let X^i be the ith $k \times r$ block of variables and define the polynomial $p^i(X)$ by $\lambda(X^i)$. The polynomial $p(X) = \sum_{i=1}^{b} p^i(X)$ counts the number of X^i for which $g(X^i) = 1$ and since f is a symmetric function, $p(X)$ determines $(f \circ g)(X)$. Now apply Theorem 1 to p with $D = \{0, 1\}$.

Main Idea for our Protocol

As mentioned earlier, a polynomial p of degree less than k can be evaluated by k players in the NOF model by decomposing p as a sum of k polynomials, where the i-th polynomial can be evaluated privately by Player i. For a polynomial of degree k or more we can't do this. Still every polynomial p can be decomposed as a sum of polynomials $q_0 + q_1 + \cdots + q_k$ where q_0 consists of monomials that depend on every row of A (and thus can't be evaluated by any one player) and q_i consists of all monomials that contain at least one variable from each of the rows $1, \ldots, i-1$ and no variable from row i, and can thus be evaluated by Player i. So the problematic part is q_0, which is identically 0 if p has degree less than k. The first (simple) idea is that we don't need q_0 to be identically 0, we only need that $q_0(A) = 0$. The second idea is to consider alternative bases (rather than the standard monomial basis) for writing polynomials. A natural set of bases to consider are *shifted* monomial bases, where we fix a matrix B and consider the B-shifted basis consisting of products of terms of the form $x_{i,j} - B_{i,j}$. Each such B gives rise to an alternative decomposition $q_0^B + \cdots + q_k^B$. A simple but key observation is that the polynomial q_0^B varies depending on B, and it suffices for the players to agree on B so that $q_0^B(A) = 0$. Furthermore for our set up, the polynomial p is initially given as a sum of polynomials p^u each depending on only a few variables per row. The players can choose a different shift B^u for each polynomial p^u and decompose p^u with respect to that basis. Hence, the problem becomes to find a way for the players to identify and agree upon a sequence $(B^u : u \in [b])$ of shift matrices such that for every $u \in [b]$ when p^u is decomposed with respect to B^u the associated polynomial q_0^u evaluated at A is 0. It turns out that, using the fact that each p^u depends on only a few variables per row, this is easy to do.

To give a simple illustration of this idea, we give a short reformulation of the proof of Gromulsz' result that $GIP_k^m(A) = \sum_{i=1}^m \prod_{j=1}^k A_{i,j}$ has a very efficient protocol if $k > 1 + \log(m)$. We work over the field $\mathbb{F}_p$ where p is a prime larger than m (which we can assume is less than $2m$), which does not change the answer.

Here's the protocol: Since $k > 1 + \log(m)$ the number of columns is less than 2^{k-1} and Player k (who sees all rows but the last row) can identify a vector v of length $k-1$ that disagrees with every column he sees. He announces this vector. The players define the $k \times m$ matrix B to have all columns equal to the vector $\bar{v}$ (the vector obtained by complementing v) followed by a 0. All players can rewrite GIP_k^m in terms of the decomposition $q_0^B + \cdots + q_k^B$ described above. For $1 \leq i \leq k$, Player i is able to evaluate $q_i^B(A)$ and announce the result, and the output of the protocol is then $\sum_{i=1}^k q_i^B(A)$. The correctness of the protocol follows from the observation that $q_0^B(A) = 0$ since the monomials appearing in q_0^B are of the form $\prod_{i=1}^k (A_i^u - B_i^u)$ and for each column of A, there is an entry that agrees with the corresponding entry of B. The total cost of the protocol is $k - 1 + k\lceil \log(p) \rceil$.

The previous protocols for composed functions by Grolmusz [12], Babai et al. [3] and Ada et al. [2] did not use this polynomial view. In his Ph. D. thesis, Ada ([1]) gave an interpretation of the protocol of [2] in terms of polynomials, but not in terms of shifted bases. The use of shifted bases is the main technical innovation of this paper.

2 Some definitions

$\mathbb{F}[x_1, \ldots, x_n]$ denotes the ring of polynomials over field $\mathbb{F}$. The set of monomials $x_1^{j_1} \ldots x_n^{j_n}$ where $(j_1, \ldots, j_n) \in \mathbb{N}^n$ is a basis. More generally, for $c = (c_1, \ldots, c_n) \in \mathbb{F}^n$ the set of *c-shifted monomials* $(x_1 - c_1)^{j_1} \ldots (x_n - c_n)^{j_n}$ comprise a basis, called the *c-shifted basis*. A polynomial p is *independent* of x_i if no monomial in the monomial expansion of p includes x_i.

In the NOF setting, the variables are $(x_{i,j} : 1 \le i \le k, 1 \le j \le m)$. An *assignment* is a $k \times m$ matrix A. A polynomial p which contains no variable of row i is said to be *independent of row i*. The *row-by-row decomposition of p relative to assignment B* expresses p as the sum $q_0^B + q_1^B + \cdots + q_k^B$, as follows. Expand p in the B-shifted basis and let q_0^B be the sum of those (shifted) monomials in the expansion (with coefficients) that depend on every row, and for $i \ge 1$ let q_i^B be the sum of all monomials that are independent of row i and dependent on rows $1, \ldots, i-1$. Note each monomial is included in one and only one of the polynomials.

3 Proof of Theorem 1

The goal is to evaluate $p(A)$. Suppose the players are all given some fixed auxiliary assignment B. All of them can compute the row-by-row decomposition $q_0^B + \cdots + q_k^B$. Player i can evaluate $q_i^B(A)$ and announce the result with total cost $k\lceil \log |\mathbb{F}| \rceil$. If it happens that $q_0^B(A) = 0$ then this is enough to determine $p(A)$. It therefore suffices to show how the players agree on a matrix B such that $q_0^B(A) = 0$.

To do this, we use the hypothesis of the theorem that $p = p^1 + \cdots + p^b$ where p^j depends on at most r variables per row. We define a simultaneous protocol Π_C which depends on a $k \times r$ matrix C. We'll show that this protocol works provided that C satisfies certain properties. We will also show that the players can agree on a C to satisfy these properties (either using shared randomness, or deterministically by having Player k choose C).

The matrix C is used to define $k \times m$ matrices $B^1(C), \ldots, B^b(C)$ as follows: For each $u \in [b]$, let X_i^u be the sequence of (at most r) variables in row i on which p^u depends. In $B^u(C)$, assign the variables of X_i^u from left to right according to row i of C. Other variables in row i are set to 0. Let $q_0^u + q_1^u + \cdots + q_k^u$ be the row-by-row decomposition of p^u relative to $B^u(C)$. Given C, the matrices $B^u(C)$ and the decomposition of p^u can be computed privately by each player.

Now in Π_C each player i announces $\alpha_i = \sum_{u=1}^b q_i^u(A)$ and the output of the protocol is $\sum_i \alpha_i$. The cost is $k\lceil \log |\mathbb{F}| \rceil$.

The difference of the output of the protocol and the correct answer is $p(A) - \sum_{u=1}^b q_0^u(A)$, so it suffices to select C so that $q_0^u(A) = 0$ for all u. The following definitions will be helpful to achieve this.

- For polynomial p^u and row index i, and for matrices A and B we write $B \equiv_{p^u,i} A$ if A and B agree on all variables of row i on which p^u depends.
- For $u \in [b]$ and $j \in [k]$ we say that C satisfies property $Q^u(j)$ if there is an index $i^u \ne j$ such that $B^u(C) \equiv_{p^u,i^u} A$.
- For $j \in [k]$ we say that C satisfies property $Q(j)$ if it satisfies $Q^u(j)$ for every $u \in [b]$.

Observe that if C satisfies property $Q(k)$, then for each $u \in [b]$ there is an index $i^u < k$ such that $B^u(C)$ agrees with A on all variables of row i^u that appear in p^u. Each $B^u(C)$-shifted monomial of q_0^u contains a variable from each row so in particular it contains a variable from row i^u and thus the monomial vanishes at A. Thus $q_0^u(A) = 0$ for all u and so Π_C will give the correct answer. Observe also that Player k is able to privately check whether a matrix C satisfies $Q(k)$.

▶ **Claim 3.** *Let $\gamma > 0$, $1 \le j \le k$ and $k \ge \ln(b/\gamma)|D|^r + 1$. If C is chosen uniformly at random from among $k \times r$ matrices with entries in D, the probability that the matrix C does not satisfy $Q(j)$ is at most γ.*

Proof. By hypothesis, p^u depends on at most r variables from row i. Thus, for $i \neq j$, the probability that $B^u(C) \equiv_{p^u,i} A$ is at least $1/|D|^r$. Hence, the probability that $Q^u(j)$ does not hold (which is the probability that for all $i \neq j$, $B^u \not\equiv_{(p^u,i)} A$), is at most $(1-1/|D|^r)^{k-1} \leq e^{-(k-1)/|D|^r}$. Taking a union bound over $u \in [b]$ gives that the probability that $Q(j)$ fails is at most $be^{-(k-1)/|D|^r} \leq be^{-(k-1)/|D|^r} \leq \gamma$ using the hypothesized lower bound on k. ◀

We now state our randomized simultaneous message protocol: players use public coins to uniformly sample C. Every player other than player k runs Π_C. Player k first checks whether C satisfies $Q(k)$ (which can be done privately). If it does then he runs Π_C and makes the appropriate announcement. If C does not satisfy $Q(k)$, Player k announces "failure". By Claim 3, assuming that $k \geq 1 + \ln(b/\gamma)|D|^r$, this happens with probability at most γ. Each player sends at most $\lceil \log |\mathbb{F}| + 1 \rceil$ bits (where the "+1" includes the possibility of failure), for a total of $k\lceil \log |\mathbb{F}+1| \rceil$ bits.

For the deterministic protocol, if we take $\gamma = 1/2$ in the Claim, then for $k \geq 1+\ln(2b)|D|^r$, there is a matrix C satisfying $Q(k)$. Player k can select such a C privately satisfying $Q(k)$ and announce it (using $kr\lceil \log |D| \rceil$ bits). The players then run Π_C. The total communication is at most $k(r\lceil \log |D| \rceil + \lceil \log |\mathbb{F}| \rceil)$.

The communication costs of these randomized and deterministic protocols grow linearly with k. To reduce the communication cost to the cost claimed in the theorem, let $k' = \lceil 1 + |D|^r \ln(b/\gamma) \rceil$. Without any communication, each player $1, \ldots, k'$ can simplify the polynomial p by substituting in the variables appearing in rows after k'. This gives a polynomial p' that depends only on the first k' rows. The polynomial p' and the number k' satisfy the hypotheses for the above arguments for both the randomized and deterministic protocols. So players $1, \ldots k'$ can evaluate p' with the rest of the players remaining silent. Thus, replacing k by k' in the cost of the protocols above, completely establishes Theorem 1.

4 Conclusion and Open Problems

We give the first efficient NOF protocol for composed functions of block width greater than 1. Some further questions suggested by our work are stated below:

- To de-randomize our simultaneous message protocol, we used interaction in a very limited way. Can it be made a simultaneous deterministic protocol? The protocol Π_C is simultaneous, so the non-simultaneity only comes from having to choose C satisfying Claim 3. In our protocol this is done by Player k but it seems possible that this can be done simultaneously. Player j can privately determine the set of all matrices C that satisfy $Q(j)$. Claim 3 can be easily modified to show that (for k a bit larger than $2^r + \ln(b)$) there are several matrices C that satisfy $Q(i)$ for all i. Consider the simultaneous protocol in which each player j announces every C that satisfies $Q(j)$ together with his announcement for the protocol Π_C. For C that satisfies $Q(j)$ for all j, the players will have all run Π_C from which $p(A)$ can be deduced. The problem with this protocol is that if there are many matrices that satisfy $Q(j)$ for some j then it may be very costly. This gives rise to the following problem: is it possible for each player j to (privately) select a small subset $\mathcal{C}_j$ of matrices satisfying $Q(j)$ in such a way that $\cap_j \mathcal{C}_j$ is non-empty. If so, then Player j can announce only those matrices in $\mathcal{C}_j$, thereby giving an efficient simultaneous NOF protocol.
- Our protocol works for all inner functions of block width r. The number of players and the communication needed is exponential in r. Can the dependence on r be improved? The only lower bound on the communication we know is linear in r, which comes from a

simple counting argument (which is essentially the same argument which shows that for general functions on mk variables there is a function that requires communication $\Omega(m)$.)

- If we restrict the inner function to a specific interesting function, such as $T_t^{k,r}$, then the counting lower bounds don't work. Are there protocols that handle larger block width for this function?

Acknowledgements. We are grateful to anonymous referees for their many valuable comments and suggestions that helped improve readability of the paper. In particular, it has resulted in a simplification of the proof of Theorem 1.

References

1 A. Ada. *Communication complexity.* PhD thesis, McGill University, 2013.

2 A. Ada, A. Chattopadhyay, O. Fawzi, and P. Nguyen. The NOF multiparty communication complexity of composed functions. In *International Colloquium on Automata, Languages and Programming (ICALP)*, pages 13–24, 2012.

3 L. Babai, A. Gál, P. G. Kimmel, and S. V. Lokam. Communication complexity of simultaneous messages. *SIAM J. of Computing*, 33:137–166, 2003.

4 L. Babai, P. G. Kimmel, and S. V. Lokam. Simultaneous messages vs. communication. In *12th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 361–372. Springer, 1995.

5 L. Babai, N. Nisan, and M. Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45(2):204–232, 1992.

6 P. Beame, T. Pitassi, N. Segerlind, and A. Wigderson. A strong direct product theorem for corruption and the multiparty communication complexity of disjointness. *Computational Complexity*, 15(4):391–432, 2006.

7 Richard Beigel and Jun Tarui. On ACC. *Computational Complexity*, 4:350–366, 1994.

8 A. K. Chandra, M. L. Furst, and R. J. Lipton. Multiparty protocols. In *15th ACM Symposium on Theory of Computing (STOC)*, pages 94–99, 1983.

9 A. Chattopadhyay and A. Ada. Multiparty communication complexity of disjointness. Technical Report TR08-002, Electronic Colloquium on Computational Complexity (ECCC), 2008.

10 A. Chattopadhyay, A. Krebs, M. Koucký, M. Szegedy, P. Tesson, and D. Thérien. Languages with bounded multiparty communication complexity. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 500–511, 2007.

11 M. David, T. Pitassi, and E. Viola. Improved separations between nondeterministic and randomized multiparty communication. *ACM Transactions on Computation Theory (TOCT)*, 1(2), 2009.

12 V. Grolmusz. The BNS lower bound for multi-party protocols is nearly optimal. *Information and Computation*, 112:51–54, 1994.

13 Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129, 1991.

14 T. Lee and A. Shraibman. Disjointness is hard in the multiparty number-on-the-forehead model. *Computational Complexity*, 18(2):309–336, 2009.

15 A. A. Razborov. Lower bounds on the size of bounded-depth networks over a complete basis with logical addition. *Math. Notes of the Acad. of Sci. of USSR*, 41(3):333–338, 1987.

16 A. A. Sherstov. The multiparty communication complexity of set disjointness. In *44th Symposium on Theory of Computing (STOC)*, 2012.

17 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *19th Annual ACM Symposium on Theory of Computing*, pages 77–82. ACM Press, 1987.

18 E. Viola and A. Wigderson. Norms, XOR lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(1):137–168, 2008.

19 Ryan Williams. Non-uniform ACC circuit lower bounds. In *IEEE Conference on Computational Complexity*, pages 115–125, 2011.

On Reconstructing a Hidden Permutation

Flavio Chierichetti[1], Anirban Dasgupta[2], Ravi Kumar[3], and Silvio Lattanzi[4]

1 **Sapienza University, Rome, Italy**
flavio@chierichetti.name
2 **IIT Gandhinagar, Gandhinagar, India**
anirbandg@iitgn.ac.in
3 **Google, Mountain View, USA**
ravi.k53@gmail.com
4 **Google, New York, USA**
silviol@google.com

Abstract

The Mallows model is a classical model for generating noisy perturbations of a hidden permutation, where the magnitude of the perturbations is determined by a single parameter. In this work we consider the following reconstruction problem: given several perturbations of a hidden permutation that are generated according to the Mallows model, each with its own parameter, how to recover the hidden permutation? When the parameters are approximately known and satisfy certain conditions, we obtain a simple algorithm for reconstructing the hidden permutation; we also show that these conditions are nearly inevitable for reconstruction. We then provide an algorithm to estimate the parameters themselves. En route we obtain a precise characterization of the swapping probability in the Mallows model.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Mallows model, Rank aggregation, Reconstruction

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.604

1 Introduction

The Mallows model [16] is a classical exponential model for generating random perturbations of a fixed but hidden permutation. In this model, the perturbation noise is determined by a single parameter, which induces a distribution on the space of all permutations. The magnitude of the perturbation is measured by the Kendall tau distance, which is the number of pairwise disagreements between two permutations. When the parameter is large, the induced distribution is highly concentrated (in terms of the Kendall distance) around the hidden permutation whereas when the parameter is close to zero, the distribution is essentially uniform on all permutations. The model can be though of as a Gaussian-like distribution on permutations but with less nice properties. It easy to see that the permutation that maximizes the likelihood under the Mallows model is in fact the hidden permutation [25].

In a typical setting, the perturbations of an underlying latent permutation are modeled using a Mallows model and the goal is to reconstruct the hidden permutation using a few (independent) perturbed samples. For example, consider the problem of (inferring the hidden true) restaurant ranking in a neighborhood. If we assume that the user behavior corresponds to a Mallows model, then by using the individual restaurant rankings of a few users, one can hope to reconstruct the true ranking. Ever since its introduction more than half a century ago, the Mallows model has been extensively studied in diverse areas including statistics,

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 604–617

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

machine learning, information retrieval, combinatorics, and social choice theory. Many of the reconstruction methods used in practice are often based on heuristics with no provable guarantees or on a careful but exhaustive search.

Even though the Mallows model is a simple and elegant way to model the perturbations of permutation, it many settings, it is oversimplified. In the above example, the classical setting assumes that each user has the same noise parameter. A more realistic setting is the following. Each user comes with his/her own noise parameter that determines how much they perturb the true ranking: a conformist might have a small noise parameter whereas a maverick might choose to vastly differ from the true ranking and hence might have a bigger noise parameter [24]. Thus, it becomes important to study the Mallows reconstruction problem where each sample perturbation is generated with a possibly *different* noise parameter.

In this paper we consider this reconstruction problem for permutations on n elements, where each permutation is generated by a Mallows model with its own parameter. We first show that perfect reconstruction is achievable (with high probability) in polynomial time if the parameters are (approximately) known and their Euclidean norm is $\Omega(\log n)$. In contrast, we show that such a reconstruction is information-theoretically impossible if this norm is smaller than a constant. En route, we obtain a precise characterization of the probability of swapping the order of two elements in a permutation generated by the Mallows model. We then complement the reconstruction algorithm, which requires the parameters or their approximations to be explicitly given, by providing an algorithm that estimates the parameters. By using these two algorithms together, we can show for instance that if there are at least $\Omega(\log n)$ parameters that are more than some constant, then reconstruction is possible even if the parameters are *not* explicitly given. We also consider the setting of approximate reconstruction and provide upper and lower bounds in terms of the parameters.

There has been some theoretical work on the Mallows reconstruction problem, especially by Braverman and Mossel [3], who considered reconstruction in the classical setting. Our work, however, is different from theirs in a few ways. First, we go beyond the classical case, i.e., we do not require all the parameters to be equal to each other. Second, we give approximate reconstruction bounds that can guarantee an arbitrarily small maximum displacement, while they can only guarantee a maximum displacement of at least $\Omega(\log n)$. Third, in the classical case their algorithm requires super-polynomial time if the parameter is $o(1/\sqrt{\log n})$, while ours runs in polynomial time for any choice of the parameter.

Combinatorial properties of the permutations generated in the Mallows model have been studied in the past. The partition function, mean, and variance of the model were computed by Diaconis and Ram [5] and Starr [22]. Tail bounds on the displacement of an element in a Mallows permutation was studied by Braverman and Mossel [3] and Gnedin and Olshanski [11]; these bounds were further tightened in a very recent work by Bhatnagar and Peled [2]. The latter also studied the length of the longest increasing subsequence in a Mallows permutation, improving upon the earlier work of Mueller and Starr [19]. Finding the maximum likelihood permutation is equivalent to the well-known rank aggregation problem. This is in general NP-hard [1, 7] and has a polynomial-time approximation scheme [13].

The Mallows model has also been generalized in a different way by generalizing the Kendall distance to weigh the number of inversions with respect to each element differently [9, 10, 8]; Meila et al. [18] studied the inference problem in this model. Qin et al. [21] defined a coset-permutation distance based model that generalizes the Mallows model to general distances and yet remains computationally efficient. A few other generalizations have also been studied in machine learning; see [15, 14]. Mukherjee [20] studied the consistency of likelihood estimators of the parameters.

2 Preliminaries

Let $[n] = \{1, \ldots, n\}$ be the universe of n elements and let S_n be the symmetric group on $[n]$. Permutations in S_n are denoted by Greek symbols. For a permutation σ and an element i, let $\pi(i)$ denote the *position* (or the *rank*) of the ith element. For two permutations π and σ, let $\kappa(\pi, \sigma)$ denote the *Kendall tau* distance (the number of inversions) between them.

Let $\beta \in (0, \infty)$ be a parameter and let $\sigma \in S_n$ be a fixed permutation. In the *Mallows* model $\mathcal{M}(\sigma, \beta)$ of generating permutations [16], the parameter β and the permutation σ induce a distribution on S_n as follows:

$$\Pr_{\mathcal{M}(\sigma,\beta)}[\pi] = \frac{e^{-\beta \cdot \kappa(\pi,\sigma)}}{Z_\beta},$$

where Z_β is the normalization constant defined as $Z_\beta = \prod_{j \le n} \frac{1-e^{-\beta j}}{1-e^{-\beta}}$. We use $\pi \sim \mathcal{M}(\sigma, \beta)$ to denote that π is generated according to $\mathcal{M}(\sigma, \beta)$. Clearly, as $\beta \to 0$, the distribution gets closer to uniform on S_n and as $\beta \to \infty$, the distribution becomes more concentrated around σ. In the classical Mallows reconstruction problem, the goal is to recover σ, given a set $\{\pi_i\}$ of independent samples where each $\pi_i \sim \mathcal{M}(\sigma, \beta)$; the algorithm may or may not know β and the goal is to use as few samples as possible.

In a generalization of the Mallows model, there are m parameters $\beta_1, \ldots, \beta_m$ where each $\beta_u \in (0, \infty)$ and a fixed permutation $\sigma \in S_n$. In the corresponding reconstruction problem, given independent samples $\pi_1, \ldots, \pi_m$ where each $\pi_u \sim \mathcal{M}(\sigma, \beta_u)$, the goal is to reconstruct σ. The algorithm may or may not know the β_u's. Note the two key differences from the classical setting: (i) each sample is generated by a *different* noise parameter and (ii) exactly *one* sample is produced for each parameter. If we assume σ to be the identity permutation, we denote the Mallows model simply by $\mathcal{M}(\beta)$ and the Kendall tau metric by $\kappa(\pi) = \kappa(\sigma, \pi)$.

Let $[p]$ denote 1 if the binary predicate p holds and 0 otherwise. We use the following form of tail inequality [12]:

▶ **Theorem 1** (Hoeffding's inequality). *If $X_1, \ldots, X_n$ are independent r.v.'s, with $\ell_i \le X_i \le u_i$, then*

$$\Pr\left[\sum_i X_i \le E[X] - \lambda\right] \le \exp\left(-\frac{2\lambda^2}{\sum_i (u_i - \ell_i)^2}\right).$$

3 Swapping Probability

In this section we precisely characterize the probability that two elements are out of order in the Mallows model. We express this probability in terms of the parameter β and the distance between the two elements. For the remainder of this section, without loss of generality, we assume that σ is the identity permutation.

Let $\pi \sim \mathcal{M}(\beta)$. For $1 \le i \le n - k$, let

$$s_{\beta,k,i} = \Pr_{\pi \sim \mathcal{M}(\beta)}[\pi(i) > \pi(i+k)],$$

i.e., the probability that the ordering of the elements i and $i + k$ is not preserved. The following result [2] shows that $s_{\beta,k,i}$ is independent of i.

Let $I = (i_1, \ldots, i_k)$ be an increasing sequence of indices. For a permutation π, let $\pi_I \in S_k$ denote the induced relative ordering of π when restricted to the indices in I. For an integer b, let $I + b$ denote $(i_1 + b, \ldots, i_k + b)$.

▶ **Lemma 2** (Translation invariance [2]). *Let $I = (i_1, i_2, \ldots, i_k)$ be an increasing sequence and $\pi \sim \mathcal{M}(\beta)$. Then for any integer $1 \leq b \leq n - i_k$, π_I and π_{I+b} have the same distribution, i.e., for any $\omega \in S_k$, $\Pr[\pi_I = \omega] = \Pr[\pi_{I+b} = \omega]$.*

Using this it is easy to see that $s_{\beta,k,i}$ is independent of i, henceforth we denote this probability as $s_{\beta,k}$. We now obtain an exact expression for it.

▶ **Lemma 3.**

$$s_{\beta,k} = \frac{ke^{\beta(k+1)} + 1 - (k+1)e^{\beta k}}{\left(e^{\beta(k+1)} - 1\right)\left(e^{\beta k} - 1\right)}.$$

Proof. In order to prove the above lemma, we will use a result from [5] that describes two different insertion processes to create a Mallows permutation. The following two processes define a series of permutations $\pi_1, \ldots, \pi_n$ such that $\pi_n = \pi \sim \mathcal{M}(\beta)$. For the purpose of this proof, we use the shorthand $q = e^{-\beta}$.

Insertion process $P1$. Consider the elements $1, \ldots, n$ in this order. For each i, define π_i to be a permutation over the elements 1 to i. Define $\pi_1(1) = 1$. Also define π_i in terms of π_{i-1} as follows. First sample $\pi_i(i)$ as following:

$$\Pr[\pi_i(i) = j] = \frac{(1/q)^{j-1}}{1 + 1/q + \cdots + (1/q)^{i-1}}, \text{ for } j \in \{1, \ldots, i\}. \tag{1}$$

Then, for s such that $\pi_{i-1}(s) < \pi_i(i)$, $\pi_i(s) = \pi_{i-1}(s)$ and else $\pi_i(s) = \pi_{i-1}(s) + 1$. Finally, $\pi = \pi_n$.

Insertion process $P2$. Here, we consider the elements in the order $n, n-1, \ldots, 1$. The permutation π'_i is defined as a permutation over elements $n, n-1, \ldots, i$ and is defined as follows. We start with $\pi'_n(n) = 1$. The random variable $\pi'_i(i)$ is defined as

$$\Pr[\pi'_i(i) = j] = \frac{q^{j-1}}{1 + q + \cdots + q^{n-i-1}}, \text{ for } j \in \{1, \ldots, n-i\}. \tag{2}$$

Thus, for s such that $\pi'_{i+1}(s) < \pi'_i(i)$, we have $\pi'_i(s) = \pi'_{i+1}(s)$ and otherwise $\pi'_i(s) = \pi'_{i+1}(s) + 1$. Finally, $\pi = \pi'_1$.

Now, we try to compute the probability that $\pi(1) > \pi(k+1)$. Consider that the permutation π is being formed by the process $P1$.

$$\begin{aligned}
\Pr[\pi(1) > \pi(k+1)] &= \Pr[\pi_k(1) > \pi_{k+1}(k+1)] \\
&= \sum_{j=1}^{k} \Pr[\pi_{k+1}(k+1) < j \mid \pi_k(1) = j] \Pr[\pi_k(1) = j] \\
&= \sum_{j=1}^{k} \frac{(1/q)^j - 1}{(1/q)^{k+1} - 1} \Pr[\pi_k(1) = j].
\end{aligned}$$

Now, π_k is a permutation on $\{1, \ldots, k\}$ that is again distributed according to Mallows model with parameter β. If we use process $P2$ to generate it, the element 1 is inserted last, and hence the probability of $\pi_k(1) = j$ can be written as

$$\Pr[\pi_k(1) = j] = \frac{q^{j-1}}{1 + q + \cdots + q^{k-1}} = \frac{(1-q)q^{j-1}}{1 - q^k}.$$

Hence, we have

$$\begin{aligned}\Pr[\pi(1) > \pi(k+1)] &= \sum_{j=1}^{k} \frac{(1/q)^j - 1}{(1/q)^{k+1} - 1} \frac{(1-q)q^{j-1}}{1-q^k} \\ &= \frac{1-q}{q((1/q)^{k+1}-1)(1-q^k)} \sum_{j=1}^{k} (1-q^j) \\ &= \frac{(1-q)q^k}{(1-q^{k+1})(1-q^k)} \left(k - \frac{q-q^{k+1}}{1-q}\right) \\ &= \frac{q^k(q^{k+1}-q-kq+k)}{(1-q^{k+1})(1-q^k)}.\end{aligned}$$

Substituting $q = e^{-\beta}$, the proof is complete. ◀

Next, we obtain a simpler approximation of the swapping probability.

▶ **Lemma 4.** *For each $0 < \beta \le \beta'$ and $1 \le k' \le k$ such that $\beta k = \beta' k'$, we have $s_{\beta,k} \ge s_{\beta',k'}$. Moreover, if $\tau = \beta k$ for $\beta > 0, k \ge 1$, then we have*

$$\frac{1}{e^\tau + 1} \;\le\; s_{\beta,k} \;<\; \frac{\tau + e^{-\tau} - 1}{e^\tau + e^{-\tau} - 2},$$

where the lower bound occurs at $k = 1$ and the upper bound is attained in the limit as k increases.

Proof. We consider the function $f_\tau(\beta) = s_{\beta,\frac{\tau}{\beta}}$. We start by showing that its derivative with respect to β is negative in $(0, \tau]$. The derivative can be written as:

$$f'_\tau(\beta) = \frac{-\tau e^{\tau+2\beta} + (\tau + \beta\tau + \beta^2)e^{\tau+\beta} + (\tau - \beta\tau - \beta^2)e^\beta - \tau}{\beta^2 \left(1 - e^{-\tau}\right)\left(e^{\tau+\beta} - 1\right)^2}. \tag{3}$$

Since the denominator in (3) is a product of positive factors, we only need to focus on the numerator in (3), which can be rewritten as $\beta(\tau+\beta)\left(e^\tau - 1\right)e^\beta - \tau\left(e^{\tau+\beta} - 1\right)\left(e^\beta - 1\right) = X_\beta(\tau) - Y_\beta(\tau)$, where $X_\beta(\tau) = \beta(\tau+\beta)\left(e^\tau - 1\right)e^\beta$ and $Y_\beta(\tau) = \tau\left(e^{\tau+\beta} - 1\right)\left(e^\beta - 1\right)$. We will show that $X_\beta(\cdot)$ is pointwise smaller than $Y_\beta(\cdot)$ in $(0, \tau)$, thus proving that $f'_\tau(\beta)$ is negative in this range.

To prove $X_\beta(\tau) < Y_\beta(\tau)$, we express the two functions as power series in the variable τ and show that for each term in the series, the corresponding coefficients obey the inequality. We have

$$X_\beta(\tau) = \beta^2 e^\beta \tau + e^\beta \sum_{n=2}^{\infty} \frac{\beta + \beta^2/n}{(n-1)!}\tau^n \quad \text{and} \quad Y_\beta(\tau) = \left(e^\beta - 1\right)^2 \tau + e^\beta \sum_{n=2}^{\infty} \frac{e^\beta - 1}{(n-1)!}\tau^n.$$

Indeed, the ratio of coefficients corresponding to τ satisfy

$$\frac{\beta^2 e^\beta}{\left(e^\beta - 1\right)^2} = \frac{\beta^2}{e^\beta - 2 + e^{-\beta}} = \frac{\beta^2}{\sum_{i=1}^{\infty} \frac{2\beta^{2i}}{(2i)!}} = \frac{\beta^2}{\beta^2 + 2\sum_{i=2}^{\infty} \frac{\beta^{2i}}{(2i)!}} < 1,$$

and the ratio of coefficients corresponding to $\tau^n, n \ge 2$, satisfy

$$\frac{e^\beta \frac{\beta+\beta^2/n}{(n-1)!}}{e^\beta \frac{e^\beta - 1}{(n-1)!}} = \frac{\beta + \frac{\beta^2}{n}}{e^\beta - 1} = \frac{\beta + \frac{\beta^2}{n}}{\sum_{i=1}^{\infty} \frac{\beta^i}{i!}} = \frac{\beta + \frac{\beta^2}{n}}{\beta + \frac{\beta^2}{2} + \sum_{i=3}^{\infty} \frac{\beta^i}{i!}} < 1.$$

Thus, we conclude that $f_\tau(\beta)$ is decreasing in $0 < \beta \leq \tau$. The minimum is attained at $k = 1$:

$$s_{\tau,1} = \frac{e^{2\tau} + 1 - 2e^{\tau}}{(e^{2\tau} - 1)(e^{\tau} - 1)} = \frac{(e^{\tau} - 1)^2}{(e^{\tau} + 1)(e^{\tau} - 1)^2} = \frac{1}{e^{\tau} + 1}.$$

Likewise, the upper bound is achieved by the limiting value at $\beta \to 0^+$:

$$\lim_{\beta \to 0^+} s_{\beta, \frac{\tau}{\beta}} = \frac{\tau + e^{-\tau} - 1}{e^{\tau} + e^{-\tau} - 2}.$$

◀

Using this, we obtain simpler bounds on $s_{\beta,k}$ that will be useful.

▶ **Corollary 5.** *Let $\beta k = \tau$. Then,*

$$s_{\beta,k} \leq \begin{cases} 1/2 - \Theta(\tau) & \text{if } \tau = o(1), \\ 1/2 - c & \text{if } \tau = \Theta(1), \\ \tau/e^{\tau} & \text{if } \tau = \omega(1), \end{cases}$$

where $c = c(\tau)$ is a positive constant.

An interesting consequence of the bounds on $s_{\beta,k}$ is that if β is moderately large, then the hidden permutation can be guessed reasonably well. The following result was also obtained in [2, Proposition 1.9]; we give a proof only for completeness.

▶ **Corollary 6.** *If $\beta = \ln n + \ln \frac{1}{\epsilon}$, then $\Pr_{\mathcal{M}(\sigma,\beta)}[\sigma] \geq 1 - \epsilon$.*

Proof. For this value of β, any two adjacent elements in σ will swap with probability at most $e^{-\beta} = \epsilon/n$. By a union bound on all the $n - 1$ adjacent pairs, we get that the probability of no swaps is at least $1 - \epsilon$. ◀

4 Reconstruction when Parameters are Given

In this section we present an algorithm to reconstruct the hidden permutation σ, assuming we know an approximation to the noise parameters $\beta_1, \ldots, \beta_m$; let the corresponding approximations be $\hat{\beta}_1, \ldots, \hat{\beta}_m$. Let α be the approximation factor, i.e., the smallest number such that

$$\frac{\hat{\beta}_u}{\alpha} \leq \beta_u \leq \alpha \hat{\beta}_u, \text{ for all } u = 1, \ldots, m.$$

The quality of the reconstructed permutation will depend on α and the magnitude of $\beta_1, \ldots, \beta_m$; the latter should be hardly surprising since the closer is β to 0, the lesser $\mathcal{M}(\sigma, \beta)$ has information about σ (as $\beta \to 0$, $\mathcal{M}(\sigma, \beta)$ converges to the uniform distribution on S_n).

The basic step considers two elements $i \neq j$ with the promise that $|\sigma(i) - \sigma(j)| \geq k$ and aims to determine if i should be ranked above j or vice versa. Our algorithm decides this bit according to the following rule:

$$i\text{'s position} < j\text{'s position} \iff \left(\sum_{u=1}^{m} (-1)^{[\pi_u(i) > \pi_u(j)]} \cdot \min(\hat{\beta}_u, 1/k) \right) > 0. \tag{4}$$

▶ **Lemma 7.** *Let $k \geq 1$ be an integer and assume that for a large enough constant $c_1 > 0$, $\sum_{u=1}^{m} \min\left(k^2 \beta_u^2, 1\right) \geq c_1 \alpha^2 \ln 1/\delta$. If i and j are such that $|\sigma(i) - \sigma(j)| \geq k$, then the ordering of i and j determined by (4) is consistent with σ, with probability at least $1 - \delta$.*

Proof. Without loss of generality, let $\sigma(i) < \sigma(j)$. Define $X_u = (-1)^{[\pi_u(i) > \pi_u(j)]}$. We have

$$E[X_u] = \Pr_{\pi_u \sim \mathcal{M}(\sigma,\beta)}[\pi_u(i) > \pi_u(j)] - \Pr_{\pi_u \sim \mathcal{M}(\sigma,\beta)}[\pi_u(i) < \pi_u(j)].$$

Let $M_u = \min\left(\beta_u, \frac{1}{k}\right)$, $\hat{M}_u = \min\left(\hat{\beta}_u, \frac{1}{k}\right)$. By Corollary 5, we have $E[X_u] \geq c_0 k M_u$, where c_0 is a sufficiently small constant. Let $Y_u = \hat{M}_u X_u$ and $Y = \sum_{u=1}^m Y_u$. Note that $-\hat{M}_u \leq Y_u \leq \hat{M}_u$. Now,

$$E[Y] \geq c_0 k \sum_{u=1}^m \hat{M}_u M_u = \Delta.$$

If $Y > 0$, then (4) correctly identifies the ordering of i and j. We bound the probability of the incorrect event to be at most δ using Theorem 1:

$$\Pr[Y \leq 0] \leq \Pr[Y \leq E[Y] - \Delta] \leq \exp\left(-\frac{\Delta^2}{2\sum_{u=1}^m \hat{M}_u^2}\right) = \exp\left(-\frac{c_0^2 k^2 \left(\sum_{u=1}^m \hat{M}_u M_u\right)^2}{2\sum_{u=1}^m \hat{M}_u^2}\right). \tag{5}$$

We now apply a converse of the Cauchy–Schwarz inequality due to Cassel [23]: if two sequences $a = (a_1, \ldots, a_m)$, $b = (b_1, \ldots, b_m)$ of real numbers satisfy $c \leq \frac{a_u}{b_u} \leq C$ for each $u = 1, \ldots, m$, then $\langle a, b\rangle^2 \geq (c/C)||a||_2^2||b||_2^2$.

Setting $a_u = \hat{M}_u$, $b_u = M_u$, $c = \frac{1}{\alpha}$, and $C = \alpha$ and applying Cassel's inequality in (5),

$$\begin{aligned}\Pr[X \leq 0] &\leq \exp\left(-\frac{c_0^2 k^2 \alpha^{-2} \sum_{u=1}^m \hat{M}_u^2 \cdot \sum_{u=1}^m M_u^2}{2\sum_{u=1}^m \hat{M}_u^2}\right) = \exp\left(-\frac{1}{2} c_0^2 k^2 \alpha^{-2} \sum_{u=1}^m M_u^2\right) \\ &\leq \exp\left(-\frac{1}{2} c_0^2 k^2 \alpha^{-2} \cdot c_1 \alpha^2 k^{-2} \ln\frac{1}{\delta}\right) \leq \delta,\end{aligned}$$

as long as $c_1 \geq 2/c_0^2$. ◀

From Lemma 7, we can obtain the precise condition that guarantees the exact reconstruction of σ.

▶ **Theorem 8** (Exact reconstruction). *If $\sum_{u=1}^m \min\left(\beta_u^2, 1\right) \geq c\alpha^2 \ln n$ for some fixed constant c, then with probability at least $1 - n^{-\Theta(1)}$ we can reconstruct σ in polynomial time.*

Proof. We apply Lemma 7 with $k = 1$. Our condition on the β_u's guarantees that, with probability $1 - n^{-\Theta(1)}$, rule (4) correctly identifies the ordering of each pair of elements. Therefore we can use any sorting algorithm to produce σ. ◀

Let $\vec{\beta} = \langle \beta_1, \ldots, \beta_m \rangle$. We next show that for exact reconstruction, the above requirement on $\|\vec{\beta}\|^2$ is close to optimal, off only by a factor of $\log n$.

▶ **Theorem 9.** *Let $n = 2$, and let $c > 0$ be a small enough constant. If $\max \beta_u \leq c$ and $\|\vec{\beta}\|^2 \leq c$, then with probability $\Omega(1)$ we cannot reconstruct σ.*

Proof. Let $S_2 = \{\sigma, \sigma^R\}$ and let σ be the unknown permutation chosen uniformly at random in S_2. By Corollary 5, for any $u \in [m]$,

$$\Pr_{\mathcal{M}(\sigma,\beta_u)}[\sigma^R] = \frac{1}{2} - \epsilon_u,$$

with $\epsilon_u = \Theta(\beta_u)$. If $b_u = (-1)^{[\pi_u \neq \sigma]}$, then $E[b_u] = 2\epsilon_u$. The likelihood of σ given $\pi_1, \ldots, \pi_m$ is

$$X = \sum_{u=1}^{m} \ln \frac{\frac{1}{2} + b_u \cdot \epsilon_u}{\frac{1}{2} - b_u \cdot \epsilon_u} = 4 \sum_{u=1}^{m} \left((1 + O(\epsilon_u^2)) b_u \cdot \epsilon_u \right). \tag{6}$$

It is easy to see that $E[X] = \Theta(||\vec{\beta}||^2)$ and $\mathrm{Var}[X] = \Theta(||\vec{\beta}||^2)$. Since the terms of the sum in (6) are independent, and $||\vec{\beta}||^2 \leq c$ for a small enough constant $c > 0$, the probability that the likelihood of σ will be negative is at least some constant. Therefore, any algorithm will be incorrect with probability at least $\Omega(1)$. ◀

We now make another observation on reconstruction using Corollary 6.

▶ **Corollary 10.** *There exists a constant $c > 0$ such that if $||\vec{\beta}|| \geq c\alpha^2 \ln n$, then with probability $1 - n^{-\Theta(1)}$ we can reconstruct σ in polynomial time.*

Proof. If there exists one $\hat{\beta}_u$ larger than $c\alpha \ln n$, for some large enough $c > 0$, then by Corollary 6, $\pi_u = \sigma$ with high probability. Otherwise, all the $\hat{\beta}_u$'s will be smaller than $c\alpha \ln n$ and hence all the β_u's will be smaller than $c\alpha^2 \ln n$. This implies that $\sum_{u=1}^{m} \min\left(\beta_u^2, 1\right) \geq \sum_{u=1}^{m} \frac{\beta_u^2}{c\alpha^2 \ln n}$. Sincet $||\vec{\beta}||^2 \geq c^2\alpha^4 \ln^2 n$, we obtain

$$\sum_{u=1}^{m} \min\left(\beta_u^2, 1\right) \geq \frac{||\vec{\beta}||^2}{c\alpha^2 \ln n} = c\alpha^2 \ln n.$$

By applying Theorem 8, σ can be obtained with probability $1 - n^{-\Theta(1)}$ in polynomial time. ◀

5 Estimating the Parameters

In this section we deal with the problem of estimating the parameters $\beta_1, \ldots, \beta_m$. Again, without loss of generality, we assume the unknown permutation σ is the identity permutation.

Recall that for each β_u, we only have one sample permutation $\pi_u \sim \mathcal{M}(\beta_u)$. Our aim is to estimate the β_u values by looking only at the set $\{\pi_1, \ldots, \pi_m\}$. Before presenting our algorithm, we first state a result that bounds the deviation of each element from its position in the hidden permutation.

▶ **Theorem 11** ([2]). *For all $\beta > 0$,*

$$\Pr_{\pi \sim \mathcal{M}(\beta)}[|\pi(i) - i| > t] \leq 2e^{-t\beta},$$

and

$$c \cdot \min\left(\frac{e^{-\beta}}{1 - e^{-\beta}}, n - 1\right) \leq E[|\pi(i) - i|] \leq \min\left(\frac{2e^{-\beta}}{1 - e^{-\beta}}, n - 1\right),$$

for some absolute constant $c > 0$.

The expected Kendall tau distance of π can also be calculated exactly.

▶ **Corollary 12** ([4, 2]). *If $\pi \sim \mathcal{M}(\beta)$, then*

$$E[\kappa(\pi)] = \frac{ne^{-\beta}}{1 - e^{-\beta}} - \sum_{j=1}^{n} \frac{je^{-\beta j}}{1 - e^{-\beta j}}.$$

Furthermore, if $\beta > 0$, then for some constant $c > 0$,

$$c \cdot \min\left(\frac{ne^{-\beta}}{1-e^{-\beta}}, n(n-1)\right) \leq E[\kappa(\pi)] \leq \min\left(\frac{ne^{-\beta}}{1-e^{-\beta}}, n(n-1)\right);$$

if $\beta = \Theta(1)$ and $n = \Omega(1/\beta)$, then $c = 1 - o(1)$.

Our estimate $\hat{\beta}_u$ for the parameter β_u is obtained by simply looking at the pairwise distances $\kappa(\pi_u, \pi_v)$, and then using the minimum of those to estimate $\hat{\beta}_u$. Formally, $\hat{\beta}_u$ is defined as following:

$$\hat{\beta}_u = \ln\left(\frac{\tilde{k}_u + 1}{\tilde{k}_u}\right), \text{ where } \tilde{k}_u = \min_{v \in [m]} \frac{\kappa(\pi_u, \pi_v)}{n}. \tag{7}$$

In order to show that (7) gives a reasonable estimate of the β_u parameters, we first need to show that if $\pi \sim \mathcal{M}(\beta)$ and $\pi' \sim \mathcal{M}(\beta')$ are two sample permutations from two different Mallows models, then the Kendall distance between π and π' is related to a function of β and β'. For this, we first relate $\kappa(\pi, \pi')$ to $\kappa(\pi) + \kappa(\pi)$.

Define

$$c_\beta = 1 - \frac{\beta + e^{-\beta} - 1}{e^{\beta} + e^{-\beta} - 2} > \frac{1}{2}.$$

From Lemma 4 for $k = 1$ and $\beta > 0$, we get the following.

▶ **Corollary 13.** *If $i, j \in [n]$ such that $i > j$, then $\Pr_{\pi \sim \mathcal{M}(\beta)}[\pi(i) > \pi(j)] \geq c_\beta$.*

The above corollary can then be used to show the following lower bound on the expectation of the Kendall distance between any two random permutations. Note that an upper bound on $\kappa(\pi, \pi')$ in terms of $\kappa(\pi)$ and $\kappa(\pi')$ is trivial by the triangle inequality.

▶ **Lemma 14.** *If $\pi \sim \mathcal{M}(\beta)$ and $\pi' \sim \mathcal{M}(\beta')$, then $E[\kappa(\pi, \pi')] \geq c_{\beta'} E[\kappa(\pi)] + c_\beta E[\kappa(\pi')]$. In particular, for all $\beta, \beta' > 0$, $E[\kappa(\pi, \pi')] \geq (E[\kappa(\pi)] + E[\kappa(\pi')])/2$.*

Proof. For two permutations τ and τ', define the inversion vector $\mathsf{inv}(\tau, \tau')$ as

$$\mathsf{inv}(\tau, \tau')_{\tau(i)} = \sum_{j : \tau(j) < \tau(i)} \mathbb{1}[\tau'(j) > \tau'(i)].$$

Define $x = \mathsf{inv}(\sigma, \pi)$, $x' = \mathsf{inv}(\sigma, \pi')$, $w = \mathsf{inv}(\pi, \pi')$ and $z = \mathsf{inv}(\pi', \pi)$. By definition,

$$w_{\pi(i)} = \sum_{j : \pi(j) < \pi(i)} \mathbb{1}[\pi'(j) > \pi'(i)].$$

Then, $\kappa(\pi, \pi') = \sum_i w_i = \sum_i z_i$, and similarly for the others. Since π and π' are independent,

$$E[w_{\pi(i)}] = \sum_j E\left[\mathbb{1}[\pi(j) < \pi(i)]\, \mathbb{1}[\pi'(j) > \pi'(i)]\right] = \sum_j \Pr[\pi(j) < \pi(i)] \Pr[\pi'(j) > \pi'(i)],$$

and therefore,

$$E\left[\sum_i w_{\pi(i)}\right] = \sum_i \sum_{j<i} \Pr[\pi(j) < \pi(i)] \Pr[\pi'(j) > \pi'(i)] + \sum_i \sum_{j>i} \Pr[\pi(j) < \pi(i)] \Pr[\pi'(j) > \pi'(i)].$$

Now, using Corollary 13, we have that for $j < i$, $\Pr[\pi(j) < \pi(i)] \geq c_\beta$. Using the same argument, for $j > i$, $\Pr[\pi'(j) > \pi'(i)] \geq c_{\beta'}$. Hence,

$$E\left[\sum_i w_{\pi(i)}\right] \geq c_\beta \sum_i \sum_{j<i} \Pr[\pi'(j) > \pi'(i)] + c_{\beta'} \sum_i \sum_{j>i} \Pr[\pi(j) < \pi(i)].$$

The proof is completed by just noting that $\sum_i \sum_{j<i} \Pr[\pi'(j) > \pi'(i)] = E[\sum_i x'_i] = E[\kappa(\pi')]$ and $\sum_i \sum_{j>i} \Pr[\pi(j) < \pi(i)] = E[\sum_i x_i] = E[\kappa(\pi)]$. ◀

Thus, $E[\kappa(\pi, \pi')]$ is both upper and lower bounded by $E[\kappa(\pi)] + E[\kappa(\pi')]$ to within constant factors. We next show that $\kappa(\pi, \pi')$ is concentrated around its expectation. We will use the following concentration theorem (proved in [17] and expressed in this form in [6]).

▶ **Theorem 15** ([17]). *Let f be a function of n random variables $X_1, \ldots, X_n$, each X_i taking values in a set A_i, such that $E[f]$ is bounded. Assume that*

$$m \leq f(X_1, \ldots, X_n) \leq M.$$

Let $\mathcal{B}$ be any event and let c_i be maximum effect of f assuming $\mathcal{B}$, i.e.,

$$|E[f \mid \mathbf{X}_{i-1}, X_i = a_i, \mathcal{B}] - E[f \mid \mathbf{X}_{i-1}, X_i = a'_i, \mathcal{B}]| \leq c_i.$$

Then

$$\Pr[f > E[f] + t] \leq \exp\left(-\frac{2t^2}{\sum_i c_i^2}\right) + \Pr[\mathcal{B}^c],$$

and

$$\Pr[f < E[f] - t] \leq \exp\left(-\frac{t^2}{\sum_i c_i^2}\right) + \Pr[\mathcal{B}^c].$$

In order to apply the above tail bound to show that $\kappa(\pi, \pi')$ is concentrated, we will first need a result showing that each element does not move too much from its original position with high probability. Define $\Delta(\beta) = \frac{1}{\beta}\ln(5n^4)$. The following is easily obtained from Theorem 11.

▶ **Lemma 16.** *If $\pi \sim \mathcal{M}(\beta), \pi' \sim \mathcal{M}(\beta')$, and $\Delta' = \Delta(\beta') + \Delta(\beta)$, then*

$$\Pr[\,\forall i\ |\pi(i) - i| \leq \Delta' \text{ and } |\pi'(i) - i| \leq \Delta'] \geq 1 - n^{-4}.$$

Proof. By applying Theorem 11 and then taking a union bound over all positions. ◀

We next show that $\kappa(\pi, \pi')$ does not deviate from its expectation with high probability.

▶ **Lemma 17.** *If $\pi \sim \mathcal{M}(\beta'), \pi' \sim \mathcal{M}(\beta)$, and $\Delta' = \Delta(\beta') + \Delta(\beta)$, then*

$$\Pr[|\kappa(\pi, \pi') - E[\kappa(\pi, \pi')]| > 2\Delta'\sqrt{n \log n}] \leq 4n^{-4}.$$

Proof. We use the tail inequality in Theorem 15 to bound $\kappa(\pi, \pi')$. Let X_{2i} denote the random variable that contains the position $\pi(i)$ and let X_{2i+1} contain $\pi'(i)$. Let $f(X_1, \ldots, X_{2n}) = \kappa(\pi, \pi')$.

Let $\mathcal{B}$ be the event: "$\forall i, |\pi(i) - i| \leq \Delta'$ and $|\pi'(i) - i| \leq \Delta'$". Using Lemma 16, $\Pr[\mathcal{B}^c] \leq n^{-4}$. Let $\vec{x}, \vec{x}'$ denote a vector of size $n - i - 1$ and $\vec{b}$ denote a vector of size $i - 1$. Define $f_{\vec{b},c}(x_{i+1}, \ldots, x_n) = f(\vec{b}, X_i = c, x_{i+1}, \ldots, x_n)$. Since only the ith element causes different transpositions in the two cases, we have

$$|f_{\vec{b},c}(x_{i+1}, \ldots, x_n) - f_{\vec{b},c'}(x_{i+1}, \ldots, x_n)| \leq |c - c'|.$$

Using the insertion process $P1$ (Lemma 3), the probability of $X_{i+1}, \ldots, X_n$ assuming any set of values remains the same, irrespective of the exact values realized by the random variables $X_1, \ldots, X_i$. That is,

$$\begin{aligned}&\Pr[(X_{i+1}, \ldots, X_n) = \vec{x} \mid (X_1, \ldots, X_{i-1}) = \vec{b}, X_i = c] \\ &\quad = \Pr[(X_{i+1}, \ldots, X_n) = \vec{x} \mid (X_1, \ldots, X_{i-1}) = \vec{b}', X_i = c'].\end{aligned}$$

Combining these two facts, we have that

$$|E[f_{\vec{X}_{i-1},c}] - E[f_{\vec{X}_{i-1},c'}]| \leq |c - c'|,$$

where $\vec{X}_{i-1} = X_1, \ldots, X_{i-1}$. Conditioned on the event $\mathcal{B}^c$, we then have

$$|c - c'| \leq 2\Delta'.$$

Furthermore $0 \leq f \leq n^2$. Hence using Theorem 15, we have that

$$\Pr[f > E[f] + t] \leq \exp\left(-\frac{2t^2}{4n\Delta'^2}\right) + \frac{1}{n^4} \quad \text{and} \quad \Pr[f < E[f] - t] \leq \exp\left(-\frac{t^2}{4n\Delta'^2}\right) + \frac{1}{n^4}.$$

By choosing $t = 4\Delta'\sqrt{n \log n}$, we have $\Pr[|f - E[f]| > t] \leq 4n^{-4}$. ◀

Finally, we show that we can get a good estimate of β if n is large enough.

▶ **Lemma 18.** *Let $\beta_1 \geq \cdots \geq \beta_m > 0$ and let $c > 0$ be the constant in Corollary 12. If $n = \omega\left(\frac{e^{\beta_1}}{\beta_m^2 \ln(1/\beta_m)}\right)$, then for each $u > 1$, (7) returns an estimate $\hat{\beta}_u$ such that*

$$\beta_u - \ln 2 - o(1) \leq \hat{\beta}_u \leq \beta_u + \ln \frac{1}{c_{\beta_m} c} + o(1).$$

Proof. Note that (7) computes $\kappa(\pi_u, \pi_v)$ for each pair $u, v, u \neq v$. Applying Lemma 17 and taking a union bound over all pairs (u, v), with probability $1 - \frac{1}{n^2}$, the following event happens:

$$\forall u \neq v, |\kappa(\pi_u, \pi_v) - E[\kappa(\pi_u, \pi_v)]| \leq \Delta', \tag{8}$$

where $\Delta' = 2 \max_{u \in [m]} \Delta(\beta_u)$.

Since $\tilde{k}_{uv} = \kappa(\pi_u, \pi_v)$, using Lemma 14 for the lower bound and the triangle inequality for the upper bound, we have

$$c_{\beta_v} E[\kappa(\pi_u)] + c_{\beta_u} E[\kappa(\pi_v)] \leq E[\tilde{k}_{uv}] \leq E[\kappa(\pi_u)] + E[\kappa(\pi_v)]. \tag{9}$$

Since c_β is an increasing function of β for all $u > 1$, (9) implies

$$c_{\beta_m} E[\kappa(\pi_u)] \leq \min_v E[\tilde{k}_{uv}] \leq E[\kappa(\pi_u)] + E[\kappa(\pi_1)]. \tag{10}$$

Plugging in the values of the expectations from Corollary 12, (10) implies

$$nc\frac{c_{\beta_m} e^{-\beta_u}}{1 - e^{-\beta_u}} \leq \min_v E[\tilde{k}_{uv}] \leq n\left(\frac{e^{-\beta_u}}{1 - e^{-\beta_u}} + \frac{e^{-\beta_1}}{1 - e^{-\beta_1}}\right).$$

Hence for all $u > 1$,

$$nc\frac{c_{\beta_m} e^{-\beta_u}}{1 - e^{-\beta_u}} \leq \min_v E[\tilde{k}_{uv}] \leq 2n\frac{e^{-\beta_u}}{1 - e^{-\beta_u}}.$$

Now, we condition on the event described in (8). For $u > 1$, we have that

$$nc \cdot c_{\beta_m} \frac{e^{-\beta_u}}{1-e^{-\beta_u}} - 2\Delta' \leq \min_v \tilde{k}_{uv} \leq 2n \frac{e^{-\beta_u}}{1-e^{-\beta_u}} + 2\Delta'.$$

Hence, $\tilde{k}_u = \frac{\min_v \tilde{k}_{uv}}{n} \in \left[c_{\beta_m} c \frac{e^{-\beta_u}}{1-e^{-\beta_u}} - \frac{2\Delta'}{n}, \frac{2e^{-\beta_u}}{1-e^{-\beta_u}} + \frac{2\Delta'}{n}\right]$. Under the assumption that $n = \omega\left(\frac{4\Delta'}{c_{\beta_m} c} e^{\beta_1}\right)$, we have that $\tilde{k}_u \in \left[(1-o(1)) c_{\beta_m} c \frac{e^{-\beta_u}}{1-e^{-\beta_u}}, \frac{2(1+o(1))e^{-\beta_u}}{(1-e^{-\beta_u})}\right]$. Since $\hat{\beta}_u = \ln\left(\frac{\tilde{k}_u+1}{\tilde{k}_u}\right)$, the upper and lower bounds on $\hat{\beta}_u$ in the statement follows. The constraints on n boil down to saying that $n = \omega\left(\frac{\log n}{\beta_m c_{\beta_m}} e^{\beta_1}\right)$. Simplifying, $n = \omega\left(\frac{e^{\beta_1}}{\beta_m^2 \ln(1/\beta_m)}\right)$ is sufficient. ◀

An easy corollary is the following: a multiplicative reconstruction of the β_u's is possible for the β_u that are $\Theta(1)$ and there is at least one (unknown) permutation generated with a parameter that is large, and hence is close to the identity.

▶ **Corollary 19.** *If β_1 is such that $\beta_1 = \omega(\beta_u)$ for some $u > 1$, then*

$$(1+o(1)) c_{\beta_m} E[\kappa(\pi_u)] \leq \min_v E[\tilde{k}_{uv}] \leq (1+o(1)) E[\kappa(\pi_u)],$$

and hence for each $u > 1$, (7) returns an estimate $\hat{\beta}_u$ such that

$$\beta_u - o(1) \leq \hat{\beta}_u \leq \beta_u + \ln \frac{1}{c_{\beta_m} c} + o(1).$$

In particular, if $\beta_u = \Theta(1)$, then $\beta_1 = \omega(1)$ and the constants $c = 1 - o(1)$ and $c_{\beta_m} = 1 - o(1)$.

6 Approximate Reconstruction

Next, we show a result on approximate reconstruction of σ. We first show that if the sum of squares of β_ℓ is $\Omega(\ln n)$, i.e., the average is $\Omega(\frac{\ln n}{n})$, then we can learn an estimate $\hat{\sigma}$ of σ where the displacement of each element is bounded. We then show a simple lower bound that says that is $\sum_\ell \beta_\ell^2$ is really small, then we cannot recover anything meaningful.

▶ **Theorem 20** (Approximate reconstruction). *Let $k^\star = \arg\min_k \sum_{\ell=1}^m \min\left(k^2 \beta_\ell^2, 1\right) \geq c\alpha^2 \ln n$ for some fixed constant c and let $k^\star$ be known to the algorithm. Then with probability at least $1 - n^{-\Theta(1)}$ we can construct a permutation $\hat{\sigma}$ such that $|\hat{\sigma}(i) - \sigma(i)| \leq 2k^\star$ for all $i \in [n]$.*

Proof. Using (4) for every pair of elements, with probability at least $1 - n^{-\Theta(1)}$, we determine the rank of each element to within an additive error of $k^\star$, i.e., for each element i, Lemma 7 guarantees that all elements j such that $|\sigma(i) - \sigma(j)| \geq k^\star$ will be correctly compared to i. We now need to find out a feasible permutation $\hat{\sigma}$ out of this set of comparisons such that the maximum displacement in $\hat{\sigma}$ is bounded.

Define the *score* of element i to be the number of other elements such that the right hand side of (4) holds. We define the permutation $\hat{\sigma}$ as the permutation that results from sorting the elements according to this score (ascending). We show that the displacement of every element is bounded by $2k^\star$. Consider any element i. By Lemma 7, the score of element i is at least $\max(1, i - k^\star)$ and at most $\min(i + k^\star, n)$. Therefore, $\hat{\sigma}(i) \in [\max(1, i - 2k^\star), \min(i + 2k^\star, n)]$. ◀

We now show a simple lower bound for approximate reconstruction.

▶ **Theorem 21.** *Let $\epsilon > 0$ be a small enough constant and let $\sqrt{\sum_{\ell=1}^{m} \beta_\ell} \leq \epsilon/n$. If σ is chosen uniformly at random in S_n, then any $\hat{\sigma}$ that is output by any algorithm satisfies $E[\kappa(\hat{\sigma}, \sigma)] \geq \left(\frac{1}{4} - \epsilon\right) n^2$.*

Proof. Consider the probability of the generic sequence of independent samples $\pi_1, \ldots, \pi_m$:

$$\Pr[\pi_1, \ldots, \pi_m \mid \sigma] = \prod_{\ell=1}^{m} \frac{e^{-\beta_\ell \kappa(\pi_\ell, \sigma)}}{Z_{\beta_\ell}} = e^{-\sum_{\ell=1}^{m} \beta_\ell \kappa(\pi_\ell, \sigma)} \cdot \prod_{\ell=1}^{m} Z_{\beta_\ell}^{-1}. \tag{11}$$

Since for each ℓ, $0 \leq \kappa(\pi_\ell, \sigma) \leq \binom{n}{2}$, we have

$$0 \leq \sum_{\ell=1}^{m} \beta_\ell \kappa(\pi_\ell, \sigma) \leq \binom{n}{2} \sum_{\ell=1}^{m} \beta_\ell \leq \epsilon. \tag{12}$$

It follows that for each sequence of samples $\pi_1, \ldots, \pi_m$, using (11) and (12), we have

$$e^{-\epsilon} \prod_{\ell=1}^{m} Z_{\beta_\ell}^{-1} \leq \Pr[\pi_1, \ldots, \pi_m | \sigma] \leq \prod_{\ell=1}^{m} Z_{\beta_\ell}^{-1}.$$

Thus, the probabilities of obtaining a set of m permutations are all within $e^{-\epsilon}$ factor of each other. For a set S of input permutations, let $S \sim U^m$ mean that each permutation is chosen uniformly at random, let $S \sim \mathcal{M}^m$ mean that the permutations are chosen according to the Mallows model with the parameters as in the Lemma statement, and let $\hat{\sigma}(S)$ be the solution returned by the algorithm on input S. We have

$$E[\kappa(\hat{\sigma}(S), \sigma) \mid S \sim U^m] \geq \frac{1}{2}\binom{n}{2},$$

as otherwise we can work with σ^R instead. Since under the given assumptions, the probability of obtaining each set S is within $e^{-\epsilon}$ of the uniform distribution,

$$|E[\kappa(\hat{\sigma}(S), \sigma) \mid S \sim \mathcal{M}^m] - E[\kappa(\hat{\sigma}(S), \sigma) \mid S \sim U^m]| \leq (1 - e^{-\epsilon}) E[\kappa(\hat{\sigma}(S), \sigma) \mid S \sim U^m].$$

Hence, $E[\kappa(\hat{\sigma}(S), \sigma) | S \sim \mathcal{M}^m] \geq (\frac{1}{4} - \epsilon) n^2$. ◀

▶ **Corollary 22.** *If $\hat{\sigma}$ is the output of an algorithm, then $\kappa(\hat{\sigma}, \sigma) = \Omega(n/\sqrt{\sum_{\ell=1}^{m} \beta_\ell})$.*

To interpret these lower bounds, we consider a concrete special case. Suppose $m = \omega(\log n)$ and $\beta_1 = \cdots = \beta_m = \beta$. Then, Theorem 20 guarantees a maximum element displacement of $O(\sqrt{(\log n)/(\beta^2 m)})$, which means that the total Kendall distance is $O(n\sqrt{(\log n)/(\beta^2 m)})$. On the other hand, for this setting, Theorem 21 obtains a Kendall distance lower bound of $\Omega(n\sqrt{1/(\beta m)})$. Thus, the gap between the upper bound and the lower bound is $O(\sqrt{(\log n)/\beta})$.

Acknowledgments. We thank the anonymous reviewers for their many valuable suggestions, especially towards a simpler proof of Lemma 3.

References

1 J. Bartholdi, C. A. Tovey, and M. A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6(2):157–165, 1989.

2 N. Bhatnagar and R. Peled. Lengths of monotone subsequences in a Mallows permutation. *Probability Theory and Related Fields*, To appear.

3 M. Braverman and E. Mossel. Sorting from noisy information. *CoRR, abs/0910.1191*, 2009.

4 W. Cheng and E. Hüllermeier. Instance-based label ranking using the Mallows model. In *ECCBR Workshops*, pages 143–157, 2008.

5 P. Diaconis and A. Ram. Analysis of systematic scan Metropolis algorithms using Iwahori–Hecke algebra techniques. *The Michigan Mathematical Journal*, 48(1):157–190, 2000.

6 D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomised Algorithms*. Cambridge University Press, 2009.

7 C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *WWW*, pages 613–622, 2001.

8 M. A. Fligner and J. S. Verducci. Distance based ranking models. *Journal of the Royal Statistical Society B*, 48:359–369, 1986.

9 M. A. Fligner and J. S. Verducci. Multistage ranking models. *Journal of the American Statistical Association*, 43(403):892–901, 1988.

10 M. A. Fligner and J. S. Verducci. Posterior probability for a consensus ordering. *Psychometrika*, 55:53–63, 1990.

11 A. Gnedin and G. Olshanski. The two-sided infinite extension of the Mallows model for random permutations. *Advances in Applied Mathematics*, 48(5):615–639, 2012.

12 W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

13 C. Kenyon-Mathieu and W. Schudy. How to rank with few errors. In *STOC*, pages 95–103, 2007.

14 A. Klementiev, D. Roth, and K. Small. Unsupervised rank aggregation with distance-based models. In *ICML*, pages 472–479, 2008.

15 G. Lebanon and J. Lafferty. Cranking: Combining rankings using conditional probability models on permutations. In *ICML*, pages 363–370, 2002.

16 C. L. Mallows. Non-null ranking models I. *Biometrika*, 44(1-2):114–130, 1957.

17 Colin McDiarmid. Concentration. In M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed, editors, *Probabilistic Methods for Algorithmic Discrete Mathematics*. Springer, 1998.

18 M. Meila, K. Phadnis, A. Patterson, and J. A. Bilmes. Consensus ranking under the exponential model. In *UAI*, pages 285–294, 2007.

19 C. Mueller and S. Starr. The length of the longest increasing subsequence of a random Mallows permutation. *Journal of Theoretical Probability*, pages 1–27, 2011.

20 S. Mukherjee. Estimation of parameters in non uniform models on permutations. Technical Report 1307.0978, arXiv, 2013.

21 T. Qin, X. Geng, and T-Y. Liu. A new probabilistic model for rank aggregation. In *NIPS*, pages 1948–1956, 2010.

22 S. Starr. Thermodynamic limit for the Mallows model on S_n. Technical Report 0904.0696, arXiv, 2009.

23 G. S. Watson. Serial correlation in regression analysis. I. *Biometrika*, 42(3/4):327–341, 1955.

24 P. Yin, P. Luo, M. Wang, and W-C. Lee. A straw shows which way the wind blows: Ranking potentially popular items from early votes. In *WSDM*, pages 623–632, 2012.

25 H. P. Young. Optimal voting rules. *The Journal of Economic Perspectives*, 9(1):51–64, 1995.

Two Sides of the Coin Problem*

Gil Cohen[1], Anat Ganor[1], and Ran Raz[1,2]

1 **Weizmann Institute of Science, Rehovot, Israel**
{gil.cohen,anat.ganor,ran.raz}@weizmann.ac.il
2 **Institute for Advanced Study, Princeton, NJ, USA**

Abstract

In the *coin problem*, one is given n independent flips of a coin that has bias $\beta > 0$ towards either Head or Tail. The goal is to decide which side the coin is biased towards, with high confidence. An optimal strategy for solving the coin problem is to apply the majority function on the n samples. This simple strategy works as long as $\beta > \Omega(1/\sqrt{n})$. However, computing majority is an impossible task for several natural computational models, such as bounded width read once branching programs and $\mathbf{AC^0}$ circuits.

Brody and Verbin [8] proved that a length n, width w read once branching program cannot solve the coin problem for $\beta < O(1/(\log n)^w)$. This result was tightened by Steinberger [20] to $O(1/(\log n)^{w-2})$. The coin problem in the model of $\mathbf{AC^0}$ circuits was first studied by Shaltiel and Viola [19], and later by Aaronson [1] who proved that a depth d size s Boolean circuit cannot solve the coin problem for $\beta < O(1/(\log s)^{d+2})$.

This work has two contributions:

- We strengthen Steinberger result and show that any Santha-Vazirani source with bias $\beta < O(1/(\log n)^{w-2})$ fools length n, width w read once branching programs. In other words, the strong independence assumption in the coin problem is completely redundant in the model of read once branching programs, assuming the bias remains small. That is, the exact same result holds for a much more general class of sources.
- We tighten Aaronson's result and show that a depth d, size s Boolean circuit cannot solve the coin problem for $\beta < O(1/(\log s)^{d-1})$. Moreover, our proof technique is different and we believe that it is simpler and more natural.

1998 ACM Subject Classification F.1.0 Computation by Abstract Devices – General

Keywords and phrases bounded depth circuits, read once branching programs, Santha-Vazirani sources, the coin problem

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.618

1 Introduction

In the *Coin Problem*, defined by Brody and Verbin [8], one is given n independent flips of a coin that has bias $\beta > 0$ towards either Head or Tail. The goal is to decide which side the coin is biased towards, with high confidence (say, 2/3). It is not hard to see that the best strategy for solving the coin problem is to apply the majority function on the n outcomes. By Chernoff bound, this strategy works as long as $\beta > c/\sqrt{n}$, for some large enough constant c. However, taking the majority on n bits is provably an impossible task for several natural computational models, such as bounded width read once branching programs (henceforth, ROBP) and $\mathbf{AC^0}$ circuits.

* This work was supported by an ISF grant, by the I-CORE Program of the Planning and Budgeting Committee and by NSF grant numbers CCF-0832797, DMS-0835373.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 618–629

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

The coin problem is related to two other well-studied notions of approximating the majority function. The first notion is the "promise" problem of computing majority. Namely, it asks for upper and lower bounds, in different computational models, for computing the majority function correctly only on inputs that have bias at least ε. This central problem received a considerable attention in the literature (see [2], [4], [21], [3], [9], [5], [25], [26], [13], [10] and references therein). In the second notion (see, e.g., [16], [5]) one considers functions that agree with the majority function on all but δ fraction of the inputs (regardless of their Hamming weight). The coin problem can be seen as a combination of these two notions. Intuitively, it is an easier problem to solve as it allows both types of slackness, and thus poses a greater challenge for proving lower bounds.

Motivated by the construction of pseudorandom generators for ROBP, Brody and Verbin [8] considered the coin problem for the model of ROBP with bounded width. Informally speaking, a width w ROBP is a non-uniform model of computation that gets the flip outcomes one by one in a stream, and can "remember" at most $\log_2 w$ bits of information at each point in time, concerning the past outcomes. For, say, constant w, such a model cannot compute majority.[1]

In [8] it is shown that width w length n ROBP cannot solve the coin problem for $\beta < O(1/(\log n)^w)$. This result was later tightened by Steinberger [20] to $\beta < c/(2\log n)^{w-2}$ for some constant c.

A different, yet essentially equivalent formulation of the coin problem, is where given a coin that is either unbiased or has bias at least $\beta > 0$ towards Head, the goal is to distinguish between the two cases, with high confidence. As mentioned above, Brody and Verbin [8], and later on Steinberger [20], proved that a product distribution of bits, with small enough bias each, cannot be distinguished from the uniform distribution by ROBPs with bounded width. In other words, a product distribution with small enough bias *fools* ROBPs with bounded width.

Our first result shows that the independence assumption, which is necessary in many settings, is in fact completely redundant. More formally, we show that any Santha-Vazirani source, with small enough bias, fools ROBPs with bounded width. Recall that a Santha-Vazirani source [18] on n bits with bias β is a distribution $\mathbf{X}_1, \ldots, \mathbf{X}_n$, where each bit $\mathbf{X}_i$ is some adversarially chosen (probabilistic) function of $\mathbf{X}_1, \ldots, \mathbf{X}_{i-1}$, under the promise that $\mathrm{bias}(\mathbf{X}_i \mid \mathbf{X}_1 = x_1, \ldots, \mathbf{X}_{i-1} = x_{i-1}) \leq \beta$, for all prefixes $x_1, \ldots, x_{i-1}$.

Santha-Vazirani sources form a much richer class of sources than product distributions, and have been considered in the literature in various contexts (e.g., [18, 22, 23, 24, 17]). As mentioned, the original motivation of Brody and Verbin for studying the coin problem came from their approach of constructing pseudorandom generators for ROBPs. This approach yields pseudorandom generators for a natural subclass of ROBPs called *regular* ROBPs (see also, [7]). It is not clear how to construct pseudorandom generators for the non-regular case, and it is plausible that Santha-Vazirani sources are a much better starting point for such constructions. In fact, one can view the proof of Braverman *et al.* [7] as approximating Santha-Vazirani sources by recursively applying the pseudorandom generator of Impagliazzo *et al.* [12].

▶ **Theorem 1** (Santha-Vazirani sources fool ROBPs, informally stated)**.** *There exists a universal constant $c > 0$ such that the following holds. Any Santha-Vazirani source on n bits with bias $\beta < c/(2\log n)^{w-2}$ fools length n, width w ROBPs.*

[1] In this context, it is interesting to note that the classical result of Barrington [6] states that without the read-once requirement, width 5 is sufficient for computing majority.

The proof of the above theorem is based on a reduction to the result of Steinberger [20]. We move on to present our second result. As a matter of fact, the coin problem was studied by Shaltiel and Viola [19] and later by Aaronson [1] even prior to the work of Brody and Verbin. The motivation for studying the coin problem in each of these papers was completely different. While Brody and Verbin were motivated by the study of pseudorandom generators for ROBPs, Shaltiel and Viola considered the problem of hardness amplification, and Aaronson studied the seemingly unrelated problem of obtaining an exponential oracle separation between **BQP** and **PH**.

Aaronson ([1], Corollary 12) proved that any depth d Boolean circuit on n inputs that distinguishes a fair coin from an ε-biased coin, with constant confidence, must have size exponential in $(1/\varepsilon)^{1/(d+2)}$. A similar result is implicit in [19].

The proof strategy of Shaltiel and Viola was to transform any circuit that distinguishes a fair coin from a coin with bias ε, to a circuit that computes majority on $\Omega(1/\varepsilon)$ inputs, to which standard lower bounds apply [11, 14]. Taking a similar strategy, to prove his lower bound, Aaronson shows that a circuit that solves the coin problem can be transformed into a circuit that accepts all n bit strings with Hamming weight $n/2+1$ while rejecting all strings with Hamming weight $n/2$. Again, by [11, 14], the latter task is known to require large bounded depth circuits. In the reduction, both papers make use of depth 3 circuits for the problem of approximate majority [2], [25].

Our second contribution is an improvement over Aaronson's result. We give a tight lower bound for the size of a depth d circuit that solves the coin problem.

▶ **Theorem 2** (Coin Problem for $\mathbf{AC^0}$, informally stated)**.** *There exists a universal constant $c > 0$ such that the following holds. A depth d, size s Boolean circuit on n inputs cannot solve the coin problem for*

$$\beta < \frac{1}{(c \log s)^{d-1}}.$$

This is tight up to the multiplicative constant c [5].

Moreover, our proof technique is different and we believe that it is simpler and more natural, and it gives the tight bound. The intuition is the following: suppose one applies, say, 10β random restriction to the input (see Section 2.3 for the precise definition). Then, one expects that a function f that solves the coin problem for bias β, applied to the resulting restricted input, should not be constant with high probability, whereas by known results [11], [14] such random restriction typically collapses a bounded depth circuit to some constant. The formal proof formalizes this by expressing the confidence of f in terms of the Fourier spectrum of a random restriction applied to f (see Lemma 14), where the restriction parameter is related to the bias of the coin.

In this context, it is interesting to mention the work of Viola [26], who showed that a depth d, size s circuit can compute majority under the promise that the input has bias $\Omega(1/(\log s)^{d-3})$, and proved that this is tight. Moreover, Viola proved that a *randomized* circuit with depth d and size s can compute majority under the promise that the input has bias $\Omega(1/(\log s)^{d-1})$, and again, he proved that this bound is tight. We note however that the coin problem is an easier problem to solve, since the circuit may err on many inputs, and thus proving lower bounds is potentially more challenging.

2 Preliminaries

It will be convenient for us to think about coins with sides $\{\pm 1\}$. The bias of a $\{\pm 1\}$ random variable X, denoted by $\mathrm{bias}(X)$, is defined as $\frac{1}{2} \cdot |\mathbf{Pr}[X = 1] - \mathbf{Pr}[X = -1]|$.

▶ **Definition 3.** Let $\varepsilon \in [0, \frac{1}{2}]$. Define the product distribution $\mathbf{X}_\varepsilon^n$ supported on $\{\pm 1\}^n$ as follows. For $x \sim \mathbf{X}_\varepsilon^n$ it holds that $\mathbf{Pr}[x_i = 1] = \frac{1}{2} + \varepsilon$ (and thus $\mathbf{Pr}[x_i = -1] = \frac{1}{2} - \varepsilon$) for all $i \in [n]$.

We note that the uniform distribution over $\{\pm 1\}^n$, denoted by $\mathbf{U}^n$, is the same as $\mathbf{X}_0^n$. When n is clear from context we omit the superscript and write $\mathbf{X}_\varepsilon$ and $\mathbf{U}$.

▶ **Definition 4** (Santha-Vazirani Sources)**.** A distribution $\mathbf{X}$ supported on $\{\pm 1\}^n$ is called a *Santha-Vazirani source* with bias ε, if for every $i \in [n]$ and every $x_1, \ldots, x_n \in \{\pm 1\}$, it holds that

$$\text{bias}\left(\mathbf{X}_i \mid \mathbf{X}_1 = x_1, \mathbf{X}_2 = x_2, \ldots, \mathbf{X}_{i-1} = x_{i-1}\right) \leq \varepsilon.$$

▶ **Definition 5.** For a function $f\colon \{\pm 1\}^n \to \{\pm 1\}$ and a distribution $\mathbf{D}$ supported on $\{\pm 1\}^n$, the distinguishability of $\mathbf{D}$ from $\mathbf{U}$ by f is given by

$$\mathsf{Distinguishability}(f, \mathbf{D}) = \left|\mathbb{E}[f(\mathbf{D})] - \mathbb{E}[f(\mathbf{U})]\right|.$$

For $\varepsilon \in [0, \frac{1}{2}]$, let $\mathsf{Distinguishability}(f, \varepsilon)$ denote $\mathsf{Distinguishability}(f, \mathbf{X}_\varepsilon)$.

2.1 Read Once Branching Programs

A *branching program* of length n and width w is a directed (multi-) graph with n layers $V_0, \ldots, V_{n-1}$ of w nodes each, called states, and a final layer V_n with two nodes, accept and reject. The branching program has a designated start node on layer V_0. For every internal node (that is, nodes in layers $V_0, \ldots, V_{n-1}$), there are exactly 2 edges going out of it and both these edges go to nodes on the next layer of the branching program. One of these edges is labeled by 1 and the other is labeled by -1. There are no edges going out of the accept and reject nodes. The computation of a branching program of length n on a string $x = x_1, \ldots, x_n \in \{\pm 1\}^n$ is defined in the natural way, by following the edge labeled x_i at step i, starting from the start node. The computation accepts x if it reaches the accept state and rejects otherwise. This branching program just described is a read-once branching program, since each character of x is examined exactly once.

For a branching program f, an internal state s and $b \in \{\pm 1\}$, let $s_f(b)$ denote the state reached by following the edge labeled b going out of s in f. When f is clear from context we write $s(b)$ instead of $s_f(b)$. For a string $x \in \{\pm 1\}^n$ we define the output of f to be $f(x) := -1$ if f accepts x and $f(x) := 1$ otherwise. For any state s, let $\mathcal{R}_{f,x}(s)$ denote the event that the computation of f on input x reaches s. When f and x are clear from context we write $\mathcal{R}(s)$ instead of $\mathcal{R}_{f,x}(s)$. Hence, for a distribution $\mathbf{X}$ over $\{\pm 1\}^n$, $\mathbb{E}[f(\mathbf{X}) \mid \mathcal{R}(s)]$ is the expected output of f on input $\mathbf{X}$, conditioned on the event that the computation reaches s, and $\mathbf{Pr}[\mathbf{X}_i = b \mid \mathcal{R}(s)]$ for $b \in \{\pm 1\}$ and $i \in [n]$, is the probability that $\mathbf{X}_i = b$ conditioned on the event that f on input $\mathbf{X}$ reaches s. To simplify notation, for any two states s_1, s_2, let $\mathcal{R}(s_1, s_2)$ denote the event that the computation of f on input x reaches both s_1 and s_2.

2.2 Bounded Depth Circuits

We consider circuits consisting of unbounded fan-in AND, OR gates applied to input variables and their negation. We only consider circuits with one output. The size of a circuit is the number of gates it contains. The depth is defined as the length of the longest path (in edges) from any input to the output. The depth of a gate g in a circuit is the depth of the sub-circuit with output gate g.

A circuit is called *layered* if for every $d \geq 1$, the inputs of any depth d gate in the circuit are the outputs of depth $d-1$ gates. A layered circuit is called *alternating* if the inputs to

an AND gate (OR gate) with depth greater than 1 are the outputs of OR gates (AND gates). By standard arguments, for any size s, depth d Boolean circuit C there exists an alternating circuit C' with size at most $d \cdot s$ and depth d that computes the same function as C. The width of a layered circuit is the maximum fan-in of the gates in the bottom layer.

2.3 Fourier Analysis

▶ **Definition 6.** For a parameter $\rho \in [0,1]$ and $x \in \{\pm 1\}^n$, define the distribution $N_\rho(x)$ supported on $\{\pm 1\}^n$ as follows. For $y \sim N_\rho(x)$, for all $i \in [n]$ independently, with probability ρ the variable y_i is being set to x_i, and with probability $1-\rho$ the variable y_i is sampled uniformly at random from $\{\pm 1\}$.

▶ **Definition 7.** Let $\rho \in [0,1]$. The *noise operator* T_ρ, acting on the set of functions $\{f : \{\pm 1\}^n \to \mathbb{R}\}$, is defined as

$$T_\rho f(x) = \mathbb{E}[f(N_\rho(x))].$$

The following well-known lemma relates the Fourier representation of $T_\rho f$ to that of f.

▶ **Lemma 8.** *For any $f : \{\pm 1\}^n \to \mathbb{R}$ and $\rho \in [0,1]$,*

$$T_\rho f(x) = \sum_{S \subseteq [n]} \rho^{|S|} \widehat{f}(S) \chi_S(x).$$

For $\rho \in [0,1]$, a *ρ random restriction* is the following probabilistic process. For each $i \in [n]$, independently, leave it unset with probability ρ, and with probability $1-\rho$ set it to ± 1 uniformly and independently at random. We denote a restriction by $(J|z)$ where $J \subseteq [n]$ is the set of indices of unset variables and $z \in \{\pm 1, *\}^n$ is the values assigned to the variables, where the variables in J are assigned the symbol $*$. More precisely, $z_i = *$ if and only if $i \in J$, and otherwise z_i is the value assigned to the i^{th} variable.

Let $f : \{\pm 1\}^n \to \mathbb{R}$, and let $(J|z)$ be a restriction. We define the restricted function $f_{(J|z)} : \{\pm 1\}^n \to \mathbb{R}$ as follows: For $x \in \{\pm 1\}^n$, $f_{(J|z)}(x) = f(y)$, where $y \in \{\pm 1\}^n$ is defined as follows:

$$y_i = \begin{cases} x_i, & i \in J; \\ z_i, & i \notin J. \end{cases}$$

The following lemma can be found in [15].

▶ **Lemma 9.** *Let $f : \{\pm 1\}^n \to \mathbb{R}$. Let $(J|z)$ be a ρ random restriction. Then for $S \subseteq [n]$,*

$$\mathbb{E}_{(J|z)}\left[\widehat{f_{(J|z)}}(S)\right] = \rho^{|S|} \cdot \widehat{f}(S).$$

3 Santha-Vazirani Sources Fool Read Once Branching Programs

The following is the main theorem of this section, which is a formal restatement of Theorem 1.

▶ **Theorem 10.** *For any n, w such that $2 \le w \le \frac{\log n}{\log \log n}$ the following holds. Let f be a width w, length n ROBP. Then, for any $\varepsilon \in [0,1]$ and any Santha-Vazirani source $\mathbf{X}$ with bias ε,*

$$\mathsf{Distinguishability}\,(f, \mathbf{X}) \le \varepsilon \cdot (2 \log n)^{w-2} \cdot (1 + o(1)).$$

The proof of Theorem 10 is via a reduction to the lower bound for ROBP solving the coin problem given by Brody and Verbin [8] and later improved by Steinberger [20]. The following theorem is an adjustment of the lower bound of [20] (see Theorem 1 therein) to our notation.

▶ **Theorem 11** ([20]). *For any n, w such that $2 \leq w \leq \frac{\log n}{\log\log n}$ the following holds. Let f be a width w, length n ROBP. Then, for any $\varepsilon \in [0,1]$,*

$$\mathsf{Distinguishability}(f,\varepsilon) \leq \varepsilon \cdot (2\log n)^{w-2} \cdot (1+o(1)).$$

The following lemma, which formalizes the reduction, together with Theorem 11 complete the proof of Theorem 10.

▶ **Lemma 12.** *Let f be a width w, length n ROBP. Let $\mathbf{X}$ be a Santha-Vazirani source with bias ε. Then, there exists a width w, length n ROBP g such that*

$$\mathsf{Distinguishability}\,(f,\mathbf{X}) \leq \mathsf{Distinguishability}(g,\varepsilon). \tag{1}$$

Proof. Assume, without loss of generality, that $\mathbb{E}[f(\mathbf{U})] \leq \mathbb{E}[f(\mathbf{X})]$. Note that we may assume that for every layer $i \in [n]$ and every internal state s on V_{i-1}, it holds that

$$\mathbf{Pr}[\mathbf{X}_i = -1 \mid \mathcal{R}(s)] \leq \mathbf{Pr}[\mathbf{X}_i = 1 \mid \mathcal{R}(s)]. \tag{2}$$

If this is not the case, we flip the i^{th} coordinate of $\mathbf{X}$ in the event that f reaches s on input $\mathbf{X}_1,\ldots,\mathbf{X}_{i-1}$. Note that $\mathbf{X}$ remains a Santha-Vazirani source with bias ε. We change f accordingly, by switching the edges going out of s in f. Doing so, the expected output of (the resulted) f, both under the uniform distribution and under (the resulted distribution) $\mathbf{X}$, does not change.

We define hybrid distributions $\mathbf{X}^{(n)},\mathbf{X}^{(n-1)},\ldots,\mathbf{X}^{(0)}$ as follows. For every $i \in [n+1]$, let $\mathbf{X}^{(i-1)}$ be a distribution where the first $i-1$ bits are distributed according to $\mathbf{X}$ and the rest of the bits are distributed according to $\mathbf{X}_\varepsilon$, independently of all other bits. Note that $\mathbf{X}^{(0)}$ is exactly $\mathbf{X}_\varepsilon$ and $\mathbf{X}^{(n)}$ is exactly $\mathbf{X}$. We define $f^{(n)} = f$ and given $f^{(i)}$ for some $i \in [n]$, we define $f^{(i-1)}$ as follows. Let $t_1,t_2,\ldots,t_{|V_i|}$ be an order of the states on layer V_i such that for every $1 \leq j < |V_i|$,

$$\mathbb{E}[f^{(i)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(t_j)] \geq \mathbb{E}[f^{(i)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(t_{j+1})].$$

We start with $f^{(i-1)} = f^{(i)}$. Fix some state s on layer V_{i-1} and let j_1, j_{-1} be the indices such that $s_{f^{(i-1)}}(1) = t_{j_1}$ and $s_{f^{(i-1)}}(-1) = t_{j_{-1}}$. If $j_1 > j_{-1}$ then we switch the edges going out of s in $f^{(i-1)}$. Clearly, the expected output of $f^{(i-1)}$ under the uniform distribution does not change. Moreover, since we change only edges that are going out of layer $i-1$, we get that

$$\mathbb{E}[f^{(i-1)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(s_{f^{(i-1)}}(-1))] \leq \mathbb{E}[f^{(i-1)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(s_{f^{(i-1)}}(1))]. \tag{3}$$

First, we analyze how the expectation under the distribution $\mathbf{X}^{(i)}$ changes when we switch from $f^{(i)}$ to $f^{(i-1)}$. By the definition of $\mathbf{X}^{(i)}$, for every $b \in \{\pm 1\}$ it holds that

$$\mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i)}) \mid \mathcal{R}(s, s_{f^{(i-1)}}(b))] = \mathbb{E}[f^{(i-1)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(s_{f^{(i-1)}}(b))]$$

and

$$\mathbf{Pr}[\mathbf{X}^{(i)}_i = b \mid \mathcal{R}(s)] = \mathbf{Pr}[\mathbf{X}_i = b \mid \mathcal{R}(s)].$$

Therefore,

$$\begin{aligned}\mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i)}) \mid \mathcal{R}(s)] &= \sum_{b\in\{\pm1\}} \mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i)}) \mid \mathcal{R}(s, s_{f^{(i-1)}}(b))] \cdot \mathbf{Pr}[\mathbf{X}^{(i)}_i = b \mid \mathcal{R}(s)] \\ &= \sum_{b\in\{\pm1\}} \mathbb{E}[f^{(i-1)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(s_{f^{(i-1)}}(b))] \cdot \mathbf{Pr}[\mathbf{X}_i = b \mid \mathcal{R}(s)].\end{aligned}$$

In the same way,

$$\begin{aligned}\mathbb{E}[f^{(i)}(\mathbf{X}^{(i)}) \mid \mathcal{R}(s)] &= \sum_{b\in\{\pm 1\}} \mathbb{E}[f^{(i)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(s_{f^{(i)}}(b))] \cdot \mathbf{Pr}[\mathbf{X}_i = b \mid \mathcal{R}(s)] \\ &= \sum_{b\in\{\pm 1\}} \mathbb{E}[f^{(i-1)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(s_{f^{(i)}}(b))] \cdot \mathbf{Pr}[\mathbf{X}_i = b \mid \mathcal{R}(s)].\end{aligned}$$

When we switch from $f^{(i)}$ to $f^{(i-1)}$, we ensure that Equation (3) holds, and thus, assuming that Equation (2) also holds, the expectation can only increase. That is,

$$\mathbb{E}[f^{(i)}(\mathbf{X}^{(i)}) \mid \mathcal{R}(s)] \leq \mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i)}) \mid \mathcal{R}(s)]. \tag{4}$$

Next, we analyze how the expectation of $f^{(i-1)}$ changes when we switch from $\mathbf{X}^{(i)}$ to $\mathbf{X}^{(i-1)}$. By the definition of $\mathbf{X}^{(i-1)}$, for every $b \in \{\pm 1\}$ it holds that

$$\mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i-1)}) \mid \mathcal{R}(s, s_{f^{(i-1)}}(b))] = \mathbb{E}[f^{(i-1)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(s_{f^{(i-1)}}(b))]$$

and

$$\mathbf{Pr}[\mathbf{X}_i^{(i-1)} = b \mid \mathcal{R}(s)] = \mathbf{Pr}[(\mathbf{X}_\varepsilon)_i = b \mid \mathcal{R}(s)].$$

Therefore,

$$\mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i-1)}) \mid \mathcal{R}(s)] = \sum_{b\in\{\pm 1\}} \mathbb{E}[f^{(i-1)}(\mathbf{X}_\varepsilon) \mid \mathcal{R}(s_{f^{(i-1)}}(b))] \cdot \mathbf{Pr}[(\mathbf{X}_\varepsilon)_i = b \mid \mathcal{R}(s)].$$

Since $\mathbf{Pr}[\mathbf{X}_i = 1 \mid \mathcal{R}(s)] \leq \frac{1}{2} + \varepsilon = \mathbf{Pr}[(\mathbf{X}_\varepsilon)_i = 1 \mid \mathcal{R}(s)]$, and since Equation (3) holds, when we switch from $\mathbf{X}^{(i)}$ to $\mathbf{X}^{(i-1)}$, the expectation can only increase. That is,

$$\mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i)}) \mid \mathcal{R}(s)] \leq \mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i-1)}) \mid \mathcal{R}(s)]. \tag{5}$$

Combining Equations (4) and (5), we get that

$$\mathbb{E}[f^{(i)}(\mathbf{X}^{(i)}) \mid \mathcal{R}(s)] \leq \mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i-1)}) \mid \mathcal{R}(s)].$$

Finally, note that $\mathbf{Pr}_{\mathbf{X}^{(i-1)}}[\mathcal{R}(s)] = \mathbf{Pr}_{\mathbf{X}^{(i)}}[\mathcal{R}(s)]$. Therefore, by repeating the above arguments for every $s \in V_{i-1}$, and summing over them, we get that $\mathbb{E}[f^{(i)}(\mathbf{U})] = \mathbb{E}[f^{(i-1)}(\mathbf{U})]$ and $\mathbb{E}[f^{(i)}(\mathbf{X}^{(i)})] \leq \mathbb{E}[f^{(i-1)}(\mathbf{X}^{(i-1)})]$. Since this holds for every $i \in [n]$, we get that

$$\mathsf{Distinguishability}(f^{(n)}, \mathbf{X}^{(n)}) \leq \mathsf{Distinguishability}(f^{(0)}, \mathbf{X}^{(0)}),$$

as stated. ◀

4 The Coin Problem for $\mathbf{AC^0}$

The following is the main theorem of this section, which is a formal restatement of Theorem 2.

▶ **Theorem 13.** *Let f be a function computable by a size s, depth d Boolean circuit. Then, for all $\delta \in (0, \frac{1}{2}]$*

$$\mathsf{Distinguishability}\left(f, \frac{\delta}{(120 \cdot \log(12s/\delta))^{d-1}}\right) \leq \delta.$$

We note that this result is tight [5]. To prove Theorem 13, we start by proving a lemma that expresses the distinguishability of a function in terms of the behavior of the function under random restrictions.

▶ **Lemma 14.** *Let $f : \{\pm 1\}^n \to \{\pm 1\}$ and let $\varepsilon \in [0, \frac{1}{2}]$. If $(J|z)$ is a 2ε random restriction then*

$$\mathsf{Distinguishability}(f, \varepsilon) = \left| \mathbb{E}_{(J|z)} \left[\sum_{\emptyset \neq S \subseteq [n]} \widehat{f_{(J|z)}}(S) \right] \right| .$$

Proof. We first note that the distributions $\mathbf{X}_\varepsilon^n$ and $N_{2\varepsilon}(1^n)$ are the same. Indeed, both are product distributions, and for any $x \sim \mathbf{X}_\varepsilon^n$ and $i \in [n]$, $\mathbf{Pr}[x_i = 1] = \frac{1}{2} + \varepsilon$ by definition. On the other hand, if $x \sim N_{2\varepsilon}(1^n)$ then

$$\mathbf{Pr}[x_i = 1] = 2\varepsilon \cdot 1 + (1 - 2\varepsilon) \cdot \frac{1}{2} = \frac{1}{2} + \varepsilon.$$

Thus

$$\mathbb{E}[f(\mathbf{X}_\varepsilon^n)] = \mathbb{E}[f(N_{2\varepsilon}(1^n))].$$

According to Definition 7, we can write the RHS of the above equation as

$$T_{2\varepsilon} f(1^n) = \sum_{S \subseteq [n]} (2\varepsilon)^{|S|} \widehat{f}(S) \chi_S(1^n) = \sum_{S \subseteq [n]} (2\varepsilon)^{|S|} \widehat{f}(S),$$

where the first equality follows by Lemma 8. This, together with Lemma 9, implies that for $(J|z)$, a 2ε random restriction, we have that

$$\mathbb{E}[f(\mathbf{X}_\varepsilon^n)] = \sum_{S \subseteq [n]} \mathbb{E}_{(J|z)} \left[\widehat{f_{(J|z)}}(S) \right] = \mathbb{E}_{(J|z)} \left[\sum_{S \subseteq [n]} \widehat{f_{(J|z)}}(S) \right].$$

On the other hand,

$$\mathbb{E}[f(\mathbf{U})] = \widehat{f}(\emptyset) = \mathbb{E}_{(J|z)} \left[\widehat{f_{(J|z)}}(\emptyset) \right],$$

where the last inequality follows by Lemma 9. Thus,

$$\mathsf{Distinguishability}(f, \varepsilon) = |\mathbb{E}[f(\mathbf{X}_\varepsilon)] - \mathbb{E}[f(\mathbf{U})]| = \left| \mathbb{E}_{(J|z)} \left[\sum_{\emptyset \neq S \subseteq [n]} \widehat{f_{(J|z)}}(S) \right] \right|$$

as claimed. ◀

We also need the following well-known theorem, which is implicit in the result of [14] (see also [15], Chapter 4). For completeness, we give a proof of this theorem in Appendix A.

▶ **Theorem 15.** *For any $\delta \in (0, 1)$ the following holds. Let $f : \{\pm 1\}^n \to \{\pm 1\}$ be a function computable by an alternating circuit with size s, depth $d \geq 3$ and width w. Let $\ell = \log(\frac{2s}{\delta})$ and let $\rho = \frac{1}{10w} \cdot \left(\frac{1}{10\ell}\right)^{d-3} \cdot \frac{\delta}{10\ell}$. If $(J|z)$ is a ρ random restriction then*

$$\mathop{\mathbf{Pr}}_{(J|z)} \left[f_{(J|z)} \text{ is non-constant} \right] \leq \delta.$$

Lastly, the proof of Theorem 13 makes use of the following lemma.

▶ **Lemma 16.** *Let $\delta > 0$. Let f be a function computable by an alternating circuit with size s and depth d. Assume further that the bottom layer is an* AND *layer. Then, there exists a function g computable by an alternating circuit with size s, depth d and width $3\log(2s/\delta)$ such that for every $\varepsilon \leq 1/4$*

$$|\mathbb{E}_{x \sim \mathbf{X}_\varepsilon}[(f - g)(x)]| \leq \delta.$$

Proof of Lemma 16. Consider an alternating circuit with size s and depth d that computes f, with bottom layer consists of AND gates. By cutting all AND gates in the bottom layer with fan-in larger than $3\log(2s/\delta)$ we get a function g computable by an alternating circuit, consists of AND gates at the bottom layer, with size s, depth d and width $3\log(2s/\delta)$.

We note that $g^{-1}(1) \subseteq f^{-1}(1)$. On the other hand, consider a fan-in k AND gate that we cut. The probability, under $\mathbf{X}_\varepsilon$, that this AND gate outputs 1 is at most $(\frac{1}{2}+\varepsilon)^k$. Since we only cut AND gates with fan-in at least $3\log(2s/\delta)$

$$\left(\frac{1}{2}+\varepsilon\right)^k \leq \left(\frac{1}{2}+\varepsilon\right)^{3\log(2s/\delta)} \leq \frac{\delta}{2s},$$

where the last inequality follows by our assumption that $\varepsilon \leq \frac{1}{4}$ (which yields $\left(\frac{1}{2}+\frac{1}{4}\right)^3 < \frac{1}{2}$). Thus, by taking a union bound over all, at most s, AND gates with fan-in at least $3\log(2s/\delta)$ we get that

$$\Pr_{x\sim\mathbf{X}_\varepsilon}[f(x) \neq g(x)] \leq \frac{\delta}{2}.$$

Since f, g have range $\{\pm 1\}$ the above equation implies that $|\mathbb{E}_{x\sim\mathbf{X}_\varepsilon}[(f-g)(x)]| \leq \delta$ as stated. ◀

Proof of Theorem 13. By the assumption of the theorem, there exists a size s, depth d circuit C that computes f. By standard arguments (see Section 2.2), there exists a size $d \cdot s$, depth d alternating circuit C' that computes f. We may assume, without loss of generality, that the bottom layer of C' consists of AND gates. If this is not the case then we can replace every OR gate at the bottom layer with an AND gate applied to the negation of the literals which are wired to the original OR gate. By De Morgan's Law, it follows that the output of this new AND gate is the negation of the output of the original OR gate. We can thus continue with this process, layer by layer from bottom to top, switching the type of gates in each layer. At the end of the process we get an alternating circuit, with size $d \cdot s$ and depth d, that computes the negation of f. Clearly, a function and its negation have the same distinguishability.

By Lemma 16, there exists a function g, computable by an alternating circuit with size $d \cdot s$, depth d and width $w = 3\log(12ds/\delta)$ such that

$$|\mathbb{E}_{x\sim\mathbf{X}_\varepsilon}[(f-g)(x)]| \leq \frac{\delta}{6} \tag{6}$$

for all $\varepsilon \leq 1/4$. Let $\ell = \log(12ds/\delta)$ and let $(J|z)$ be a $\rho = \frac{1}{30\ell} \cdot \left(\frac{1}{10\ell}\right)^{d-3} \cdot \frac{\delta}{60\ell}$ random restriction. Since g is computable by a width $w = 3\ell$ alternating circuit, Theorem 15 implies that

$$\Pr_{(J|z)}\left[g_{(J|z)} \text{ is non-constant}\right] \leq \frac{\delta}{6}. \tag{7}$$

By Lemma 14,

$$\text{Distinguishability}\left(g, \frac{\rho}{2}\right) = \left|\mathbb{E}_{(J|z)}\left[\sum_{\emptyset\neq S\subseteq[n]} \widehat{g_{(J|z)}}(S)\right]\right|.$$

In the event that $g_{(J|z)}$ is a constant function, the entire Fourier mass of $g_{(J|z)}$ lies in the empty coefficient, and in such case, the sum within the expectation in the above equation

is 0. On the other hand, by Equation (7), $g_{(J|z)}$ is non-constant with probability at most $\delta/6$ and so,

$$\mathsf{Distinguishability}\left(g, \frac{\rho}{2}\right) \leq \frac{\delta}{6} \cdot \left| \mathbb{E}_{(J|z)} \left[\sum_{\emptyset \neq S \subseteq [n]} \widehat{g_{(J|z)}}(S) \;\middle|\; g_{(J|z)} \text{ is non-constant} \right] \right|.$$

Note that

$$\sum_{\emptyset \neq S \subseteq [n]} \widehat{g_{(J|z)}}(S) = g_{(J|z)}(1^n) - \widehat{g_{(J|z)}}(\emptyset),$$

which is some number in $[-2, +2]$ as $g_{(J|z)}$ has range $\{\pm 1\}$. Thus,

$$\mathsf{Distinguishability}\left(g, \frac{\rho}{2}\right) \leq 2 \cdot \frac{\delta}{6} = \frac{\delta}{3}.$$

The above equation together with Equation (6) implies that

$$\begin{aligned}
\mathsf{Distinguishability}\left(f, \frac{\rho}{2}\right) &= \left|\mathbb{E}[f(\mathbf{X}_{\rho/2})] - \mathbb{E}[f(\mathbf{U})]\right| \\
&\leq \left|\mathbb{E}[f(\mathbf{X}_{\rho/2})] - \mathbb{E}[g(\mathbf{X}_{\rho/2})]\right| + \\
&\quad \left|\mathbb{E}[g(\mathbf{X}_{\rho/2})] - \mathbb{E}[g(\mathbf{U})]\right| + \\
&\quad \left|\mathbb{E}[g(\mathbf{U})] - \mathbb{E}[f(\mathbf{U})]\right| \\
&\leq \frac{\delta}{6} + \frac{\delta}{3} + \frac{\delta}{6} < \delta.
\end{aligned}$$

Thus, by the choice of ρ we have

$$\mathsf{Distinguishability}\left(f, \frac{\delta}{(60 \cdot \log(12ds/\delta))^{d-1}}\right) \leq \delta.$$

The proof then follows since $d \leq s$. ◀

Acknowledgement. We wish to thank the anonymous referees for their helpful comments.

References

1. S. Aaronson. BQP and the Polynomial Hierarchy. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, pages 141–150. ACM, 2010.
2. M. Ajtai. Σ^1_1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24:1–48, 1983.
3. M. Ajtai. *Approximate counting with uniform constant-depth circuits*, volume 13. Amer. Math. Soc. Providence, RI, 1993.
4. M. Ajtai and M. Ben-Or. A theorem on probabilistic constant depth computations. In *Proceedings of the16th ACM Symposium on Theory of Computing*, pages 471–474. ACM, 1984.
5. K. Amano. Bounds on the size of small depth circuits for approximating majority. In *Automata, Languages and Programming*, pages 59–70. Springer, 2009.
6. D. A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *Journal of Computer and System Sciences*, 38(1):150–164, 1989.
7. M. Braverman, A. Rao, R. Raz, and A. Yehudayoff. Pseudorandom generators for regular branching programs. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 40–47. IEEE, 2010.

8 J. Brody and E. Verbin. The coin problem and pseudorandomness for branching programs. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 30–39. IEEE, 2010.

9 S. Chaudhuri and J. Radhakrishnan. Deterministic restrictions in circuit complexity. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 30–36. ACM, 1996.

10 G. Cohen, I. B. Damgård, Y. Ishai, J. Kölker, P. B. Miltersen, R. Raz, and R. D. Rothblum. Efficient multiparty protocols via log-depth threshold formulae. In *Advances in Cryptology–CRYPTO 2013*, pages 185–202. Springer, 2013.

11 J. Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual STOC*, pages 6–20, 1986.

12 R. Impagliazzo, N. Nisan, and A. Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the 26th ACM Symposium on Theory of Computing*, pages 356–364. ACM, 1994.

13 S. Kopparty and S. Srinivasan. Certifying polynomials for AC0 (parity) circuits, with applications. In *32nd International Conference on Foundations of Software Technology and Theoretical Computer Science*, page 36, 2012.

14 N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, Fourier transform and learnability. *J. ACM*, 40(3):607–620, 1993.

15 R. O'Donnell. Analyis of Boolean functions. `http://analysisofbooleanfunctions.org/`.

16 R. O'Donnell and K. Wimmer. Approximation by DNF: examples and counterexamples. In *Automata, Languages and Programming*, pages 195–206. Springer, 2007.

17 O. Reingold, S. Vadhan, and A. Wigderson. A note on extracting randomness from santha-vazirani sources. *Unpublished manuscript*, 2004.

18 M. Santha and U. V. Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of Computer and System Sciences*, 33(1):75–87, 1986.

19 R. Shaltiel and E. Viola. Hardness amplification proofs require majority. *SIAM Journal on Computing*, 39(7):3122–3154, 2010.

20 J. Steinberger. The distinguishability of product distributions by read-once branching programs. In *Computational Complexity (CCC), 2013 IEEE Conference on*, pages 248–254. IEEE, 2013.

21 L. Stockmeyer. On approximation algorithms for #P. *SIAM Journal on Computing*, 14(4):849–861, 1985.

22 U. V. Vazirani. Towards a strong communication complexity theory or generating quasi-random sequences from two communicating slightly-random sources. In *Proceedings of the 17th ACM Symposium on Theory of Computing*, pages 366–378. ACM, 1985.

23 U. V. Vazirani. Efficiency considerations in using semi-random sources. In *Proceedings of the 19th ACM Symposium on Theory of Computing*, pages 160–168. ACM, 1987.

24 U. V. Vazirani and V. V. Vazirani. Random polynomial time is equal to semi-random polynomial time. Technical report, Cornell University, 1988.

25 E. Viola. On approximate majority and probabilistic time. *computational complexity*, 18(3):337–375, 2009.

26 E. Viola. Randomness buys depth for approximate counting. In *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*, pages 230–239. IEEE, 2011.

A Proof of Theorem 15

The proof of Theorem 15 relies on Håstad Switching Lemma [11] (see also [15], Chapter 4).

▶ **Lemma 17** (Håstad switching lemma). *Let $f : \{\pm 1\}^n \to \{\pm 1\}$ be a function computable by a width w DNF or width w CNF. Let $(J|z)$ be a ρ random restriction. Then, for every $k \in \mathbb{N}$ it holds that*

$$\mathbf{Pr}\left[\mathsf{DTdepth}(f_{(J|z)}) \geq k\right] \leq (5\rho w)^k.$$

Proof of Theorem 15. Let C be an alternating circuit with size s, depth d and width w that computes f. For $i = 1, \ldots, d$ denote the number of gates at level i by s_i (and so $s_1 + \cdots + s_d = s$ and $s_d = 1$). For $j = 1, \ldots, s_2$ denote by g_j the j^{th} gate at level 2, and denote by C_j the circuit with top gate g_j. Note that all circuits $\{C_j\}_j$ are either CNF or DNF with width w. Assume without loss of generality that they are DNF.

Let $\rho_1 = \frac{1}{10w}$. Consider a ρ_1 random restriction $(J_1|z_1)$. By Håstad switching lemma (Lemma 17), for all $j \in [s_2]$,

$$\mathop{\mathbf{Pr}}_{(J_1|z_1)}\left[\mathsf{DTdepth}(C_j|_{(J_1|z_1)}) \geq \ell\right] \leq (5\rho_1 w)^\ell = 2^{-\ell},$$

and so, by union bound

$$\mathop{\mathbf{Pr}}_{(J_1|z_1)}\left[\exists j \in [s_2] \text{ such that } \mathsf{DTdepth}(C_j|_{(J_1|z_1)}) \geq \ell\right] \leq s_2 \cdot 2^{-\ell}.$$

Consider the event in which $\forall j \in [s_2]\ \mathsf{DTdepth}(C_j|_{(J_1|z_1)}) \leq \ell$. It is well known that if a function can be computed by a depth ℓ decision tree then it can be computed both by a width ℓ CNF and by a width ℓ DNF. We can therefore replace each C_j with a width ℓ CNF. Both the second and third layers in the resulted circuit consisting of AND gates. We can therefore collapse these two layers into one layer consists of s_3 AND gates. Denote by $g'_1, \ldots, g'_{s_3}$ the AND gates in the second layer of this new circuit. For $j = 1, \ldots, s_3$ denote by C'_j the width ℓ CNF with top gate g'_j.

Let $\rho_2 = \frac{1}{10\ell}$ and let $(J_2|z_2)$ be a ρ_2 random restriction. By Håstad switching lemma (Lemma 17), for $j_3 = 1, \ldots, s_3$,

$$\mathop{\mathbf{Pr}}_{(J_2|z_2)}\left[\mathsf{DTdepth}(C'_j|_{(J_2|z_2)}) \geq \ell\right] \leq (5\rho_2 \ell)^\ell = 2^{-\ell},$$

and so, by union bound,

$$\mathop{\mathbf{Pr}}_{(J_2|z_2)}\left[\exists j \in [s_3] \text{ such that } \mathsf{DTdepth}(C'_j|_{(J_2|z_2)}) \geq \ell\right] \leq s_3 \cdot 2^{-\ell}.$$

We restrict ourselves again to the event in which $\forall j \in [s_3]\ \mathsf{DTdepth}(C'_j|_{(J_2|z_2)}) \leq \ell$, use this fact to replace all C'_j with a width ℓ DNF and collapse the new second and third layer. We continue performing ρ_2 random restrictions until we are left with a depth 2 circuit, that is, either with a CNF or a DNF. Note that we perform a total of $d-3$ ρ_2 random restrictions (on top of the first ρ_1 random restriction). Denote the composed random restriction by $(J'|z')$. Then, except with probability $s \cdot 2^{-\ell}$ we end up with a width ℓ depth 2 circuit C''. We restrict ourselves to the event in which C'' has width at most ℓ.

Let $\rho_3 = \frac{\delta}{10\ell}$. Consider a ρ_3 random restriction $(J_3|z_3)$. By Håstad switching lemma,

$$\mathop{\mathbf{Pr}}_{(J_3|z_3)}\left[\mathsf{DTdepth}(C''|_{(J_3|z_3)}) \geq 1\right] \leq 5\rho_3\ell = \frac{\delta}{2}.$$

Thus, if we denote by $(J|z)$ the $\rho_1\rho_2^{d-3}\rho_3$ composed random restriction over all the random process described above, then

$$\mathop{\mathbf{Pr}}_{(J|z)}\left[f_{(J|z)} \text{ is non-constant}\right] \leq s \cdot 2^{-\ell} + \frac{\delta}{2} \leq \delta,$$

where the last inequality follows by the choice of ℓ. ◀

Absorption Time of the Moran Process*

Josep Díaz[1], Leslie Ann Goldberg[2], David Richerby[2], and Maria Serna[1]

1 **Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Spain**
2 **Department of Computer Science, University of Oxford, Wolfson Building, Parks Road, Oxford, OX1 3QD, UK**

Abstract

The Moran process models the spread of mutations in populations on graphs. We investigate the absorption time of the process, which is the time taken for a mutation introduced at a randomly chosen vertex to either spread to the whole population, or to become extinct. It is known that the expected absorption time for an advantageous mutation is $O(n^4)$ on an n-vertex undirected graph, which allows the behaviour of the process on undirected graphs to be analysed using the Markov chain Monte Carlo method. We show that this does not extend to directed graphs by exhibiting an infinite family of directed graphs for which the expected absorption time is exponential in the number of vertices. However, for regular directed graphs, we show that the expected absorption time is $\Omega(n \log n)$ and $O(n^2)$. We exhibit families of graphs matching these bounds and give improved bounds for other families of graphs, based on isoperimetric number. Our results are obtained via stochastic dominations which we demonstrate by establishing a coupling in a related continuous-time model. The coupling also implies several natural domination results regarding the fixation probability of the original (discrete-time) process, resolving a conjecture of Shakarian, Roos and Johnson.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Moran Process

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.630

1 Introduction

The Moran process [22], as adapted by Lieberman, Hauert and Nowak [17], is a stochastic model for the spread of genetic mutations through populations of organisms. Similar processes have been used to model the spread of epidemic diseases, the behaviour of voters, the spread of ideas in social networks, strategic interaction in evolutionary game theory, the emergence of monopolies, and cascading failures in power grids and transport networks [2, 3, 12, 16, 18].

In the Moran process, individuals are modelled as the vertices of a graph and, at each step of the discrete-time process, an individual is selected at random to reproduce. This vertex chooses one of its neighbours uniformly at random and replaces that neighbour with its offspring, a copy of itself. The probability that any given individual is chosen to reproduce is proportional to its *fitness*: individuals with the mutation have fitness $r > 0$ and non-mutants have fitness 1. The initial state has a single mutant placed uniformly at random in the graph,

* 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.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 630–642

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

with every other vertex a non-mutant. On any finite, strongly connected graph, the process will terminate with probability 1, either in the state where every vertex is a mutant (known as *fixation*) or in the state where no vertex is a mutant (known as *extinction*).

The principal quantities of interest are the *fixation probability* (the probability of reaching fixation) and the expected *absorption time* (the expected number of steps before fixation or extinction is reached). In general, these depend on both the graph topology and the mutant fitness. In principle, they can be computed by standard Markov chain techniques but doing so for an n-vertex graph involves solving a set of 2^n linear equations, which is computationally infeasible. Fixation probabilities have also been calculated by producing and approximately solving a set of differential equations that model the process [14]. These methods seem to work well in practice but there is no known bound on the error introduced by converting to differential equations and approximating their solution.

When the underlying graph is undirected and the mutant fitness is $r > 1$, there is a fully polynomial randomised approximation scheme (FPRAS) for computing the fixation probability [10]. The FPRAS uses the Markov chain Monte Carlo method and provides a suitable approximation in polynomial time because the expected absorption time on an n-vertex graph is at most $\frac{r}{r-1}n^4$ for $r > 1$.

1.1 Our Contributions

The main contribution of this paper is to determine the extent to which the polynomial bound on expected absorption time carries through to directed graphs. Throughout the paper, we assume that the mutant fitness r exceeds 1.

1.1.1 Regular Digraphs

We start by considering the absorption time on a strongly connected Δ-regular digraph (where every vertex has in-degree Δ and out-degree Δ). Regularity makes some calculations straightforward because the detailed topology of the graph is not relevant. We describe these first, and then discuss the more difficult questions (where topology does play a role) and state our results.

The following facts hold for Δ-regular graphs, independent of the topology.

- It is well known [17] that the fixation probability of any regular n-vertex graph is

$$\frac{1 - r^{-1}}{1 - r^{-n}} \tag{1}$$

  To see this, note that if there are k mutants, the total fitness of the population is $W_k = n + k(r-1)$. The probability that the next reproduction happens along the directed edge (u, v) is $\frac{r}{W_k}\frac{1}{\Delta}$ if u is a mutant and $\frac{1}{W_k}\frac{1}{\Delta}$, if it is not. Since the graph is Δ-regular, there are exactly as many directed edges from mutants to non-mutants as there are from non-mutants to mutants. Thus, the probability that the number of mutants increases at the next step is exactly r times the probability that it decreases, regardless of which vertices are currently mutants. Thus, the number of mutants in the population, observed every time it changes, forms a random walk on $\{0, \ldots, n\}$ with initial state 1, upward drift r and absorbing barriers at 0 and n. It is standard (e.g., [13, Example 3.9.6]) that such a random walk reaches n with the probability given by (1).
- It is also well known (e.g., [13, Example 3.9.6] or, for an approximation, [23, eq. (13)]) that the expected number of steps of this walk before absorption (which may be at either 0 or n) is a function of r and n that tends to $n(1 + \frac{1}{r})$ in the limit as $n \to \infty$, independent

of the graph structure beyond regularity. However, the number of steps taken by the random walk (often referred to as the "active steps" of the Moran process) is not the absorption time of the original process, which includes many steps at which the number of mutants does not change, either because a mutant reproduces to a mutant or because a non-mutant reproduces to a non-mutant.

In Section 4, we show that the expected absorption time of the Moran process is polynomial for regular digraphs. In contrast to the number of active steps, the absorption time does depend on the detailed structure of the graph. We prove the following upper and lower bounds, where H_n denotes the n'th harmonic number, which is $\Theta(\log n)$.

▶ **Theorem 1.** *The expected absorption time of the Moran process on a strongly connected Δ-regular n-vertex digraph G is at least $\left(\frac{r-1}{r^2}\right) nH_{n-1}$ and at most $n^2\Delta$.*

In Section 4.5, we prove the following theorem, which shows that the upper bound in Theorem 1 is tight up to a constant factor (which depends on Δ and r but not on n).

▶ **Theorem 2.** *Suppose that $r > 1$ and $\Delta > 2$. There is an infinite family $\mathcal{G}$ of Δ-regular graphs such that, when the Moran process is run on an n-vertex graph $G \in \mathcal{G}$, the expected absorption time exceeds $\frac{1}{8r}(1 - \frac{1}{r})\frac{n^2}{(\Delta-1)^2}$.*

The digraphs in the family $\mathcal{G}$ are symmetric, so can be viewed as undirected graphs. The upper bound on the expected absorption time in Theorem 1 can be improved for certain classes of regular undirected graphs using the notion of the isoperimetric number $i(G)$ of a graph G, which is defined in Section 4.4. In the full version, we prove the following theorem.

▶ **Theorem 3.** *The expected absorption time of the Moran process on a connected Δ-regular n-vertex undirected graph G is at most $2\Delta nH_n/i(G)$.*

Theorem 3 pinpoints the expected absorption time for $G = K_n$, up to a constant factor, since $i(K_n) = \lceil n/2 \rceil$ [21] and Theorem 1 gives an $\Omega(n \log n)$ lower bound. Theorem 3 is worse than the upper bound of Theorem 1 by a factor of $O(\log n)$ for the cycle C_n since $i(C_n) = 2/\lfloor n/2 \rfloor$ [21]. However, we often get an improvement by using the isoperimetric number. For example, the $\sqrt{n}$-by-$\sqrt{n}$ grid has $i(G) = \Theta(1/\sqrt{n})$ (see [8]), giving an $O(n^{3/2} \log n)$ absorption time; the hypercube has $i(G) = 1$ (see, e.g., [21]), giving an $O(n \log^2 n)$ absorption time. Bollobás [4] showed that, for every $\Delta \geq 3$ there is a positive number $\eta < 1$ such that, for almost all Δ-regular n-vertex undirected graphs G (as n tends to infinity), $i(G) \geq (1-\eta)\Delta/2$, which gives an $O(n \log n)$ absorption time since these graphs are connected.

1.1.2 Slow Absorption

Theorem 1 shows that regular digraphs, like undirected graphs, reach absorption in expected polynomial time. In Section 5 we show that the same does not hold for general digraphs. In particular, we construct an infinite family $\{G_{r,N}\}$ of strongly connected digraphs indexed by a positive integer N. We then prove the following theorem.

▶ **Theorem 4.** *Fix $r > 1$ and let $\varepsilon_r = \min(r - 1, 1)$. For any positive integer N, large enough with respect to r, the expected absorption time of the Moran process on $G_{r,N}$ is at least*

$$\frac{1}{16}\left\lfloor \left(\frac{\varepsilon_r}{32}\right)(2^N - 1)\right\rfloor \frac{\varepsilon_r}{4\lceil r \rceil + 3}.$$

Theorem 4 shows that there is an infinite family of strongly connected digraphs in which absorption time of an n-vertex graph is exponentially large, as a function of n. Thus, the techniques from [10] do not give a polynomial-time algorithm for approximating the fixation probability on digraphs.

The underlying structure of the graph $G_{r,N}$ is a large clique on N vertices and a long directed path. Each vertex of the clique sends an edge to the first vertex of the path, and each vertex of the clique receives an edge from the path's last vertex. We refer to the first N vertices of path as P and the remainder as Q. Each vertex of P has out-degree 1 but receives $4\lceil r \rceil$ edges from Q. (See Figure 1.)

Suppose that N is sufficiently large with respect to r and consider the Moran process on $G_{r,N}$. There is a reasonable probability (about $\frac{1}{4r+2}$) that the initial mutant is in the clique. The edges to and from the path have a negligible effect so it is reasonably likely (probability at least $1 - \frac{1}{r}$) that we will then reach the state where half the clique vertices are mutants. To reach absorption from this state, one of two things must happen.

For the process to reach extinction, the mutants already in the clique must die out. Because the interaction between the clique and path is small, the number of mutants in the clique is very close to a random walk on $\{0, \ldots, N\}$ with upward drift r, and the expected time before such a walk reaches zero from $N/2$ is exponential in N.

On the other hand, suppose the process reaches fixation. In particular, the vertices of P, the first part of the path, must become mutants. Note that no vertex of Q can become a mutant before the last vertex of P has done so. While all the vertices in Q are non-mutants, the edges from that part of the path to P ensure that each mutant in P is more likely to be replaced by a non-mutant from Q than it is to create a new mutant in P. As a result, the number of mutants in P is bounded above by a random walk on $\{0, \ldots, N\}$ with a strictly greater probability of decreasing than increasing. Again, this walk is expected to take exponentially many steps before reaching N.

1.1.3 Stochastic Domination

Our main technical tool is stochastic domination. Intuitively, one expects that the Moran process has a higher probability of reaching fixation when the set of mutants is S than when it is some subset of S, and that it is likely to do so in fewer steps. It also seems obvious that modifying the process by continuing to allow all transitions that create new mutants but forbidding some transitions that remove mutants should make fixation faster and more probable. Such intuitions have been used in proofs in the literature; it turns out that they are essentially correct, but for rather subtle reasons.

The Moran process can be described as a Markov chain $(Y_t)_{t\geq 1}$ where Y_t is the set $S \subseteq V(G)$ of mutants at the t'th step. The normal method to make the above intuitions formal would be to demonstrate a stochastic domination by coupling the Moran process $(Y_t)_{t\geq 1}$ with another copy $(Y'_t)_{t\geq 1}$ of the process where $Y_1 \subseteq Y'_1$. The coupling would be designed so that $Y_1 \subseteq Y'_1$ would ensure that $Y_t \subseteq Y'_t$ for all $t > 1$. However, a simple example shows that such a coupling does not always exist for the Moran process. Let G be the undirected path with two edges: $V(G) = \{1, 2, 3\}$ and $E(G) = \{(1, 2), (2, 1), (2, 3), (3, 2)\}$. Let $(Y_t)_{t\geq 1}$ and $(Y'_t)_{t\geq 1}$ be Moran processes on G with $Y_1 = \{2\}$ and $Y'_1 = \{2, 3\}$. With probability $\frac{r}{2(r+2)}$, we have $Y_2 = \{1, 2\}$. The only possible value for Y'_2 that contains Y_2 is $\{1, 2, 3\}$ but this occurs with probability only $\frac{r}{2(2r+1)}$. Therefore, any coupling between the two processes fails because

$$\Pr(Y_2 \not\subseteq Y'_2) \geq \frac{r(r-1)}{2(r+2)(2r+1)},$$

which is strictly positive for any $r > 1$. The problem is that, when vertex 3 becomes a mutant, it becomes more likely to be the next vertex to reproduce and, correspondingly, every other vertex becomes less likely. This can be seen as the new mutant "slowing down" all the other vertices.

To get around this problem, we consider a continuous-time version of the process, $\widetilde{Y}[t]$ ($t \geq 0$). Given the set of mutants $\widetilde{Y}[t]$ at time t, each vertex waits an amount of time before reproducing. For each vertex, this period of time is chosen according to the exponential distribution with parameter equal to the vertex's fitness, independently of the other vertices. (Thus, the parameter is r if the vertex is a mutant and 1, otherwise.) If the first vertex to reproduce is v at time $t + \tau$ then, as in the standard, discrete-time version of the process, one of its out-neighbours w is chosen uniformly at random, the individual at w is replaced by a copy of the one at v and the time at which w will next reproduce is exponentially distributed with parameter given by its new fitness. The discrete-time process is recovered by taking the sequence of configurations each time a vertex reproduces.

In continuous time, each member of the population reproduces at a rate given by its fitness, independently of the rest of the population whereas, in discrete time, the population has to co-ordinate to decide who will reproduce next. It is still true in continuous time that vertex w becoming a mutant makes it less likely that each vertex $v \neq w$ will be the next to reproduce. However, the vertices are not slowed down as they are in discrete time: they continue to reproduce at rates determined by their fitnesses. This distinction allows us to establish the following coupling lemma, which formalises the intuitions discussed above.

▶ **Lemma 5** (Coupling lemma). *Let $G = (V, E)$ be any digraph, let $Y \subseteq Y' \subseteq V(G)$ and $1 \leq r \leq r'$. Let $\widetilde{Y}[t]$ and $\widetilde{Y}'[t]$ ($t \geq 0$) be continuous-time Moran processes on G with mutant fitness r and r', respectively, and with $\widetilde{Y}[0] = Y$ and $\widetilde{Y}'[0] = Y'$. There is a coupling between the two processes such that $\widetilde{Y}[t] \subseteq \widetilde{Y}'[t]$ for all $t \geq 0$.*

In the paper, we use the coupling lemma to establish stochastic dominations between *discrete* Moran processes. It also has consequences concerning fixation probabilities. The fixation probability $f_{G,r}$ is the probability that the all-mutant state is reached when the Moran process is run on a digraph $G = (V, E)$, starting from a state in which a single initial mutant is placed uniformly at random. For any set $S \subseteq V$, let $f_{G,r}(S)$ be the probability of reaching fixation when the set of vertices initially occupied by mutants is S. Thus, $f_{G,r} = \frac{1}{|V|} \sum_{v \in V} f_{G,r}(\{v\})$. Using the coupling lemma, we can prove the following theorem.

▶ **Theorem 6.** *For any digraph G, if $0 < r \leq r'$ and $S \subseteq S' \subseteq V(G)$, then $f_{G,r}(S) \leq f_{G,r'}(S')$.*

This theorem has two immediate corollaries. The first was conjectured by Shakarian, Roos and Johnson [26, Conjecture 2.1]. It was known from [25] that $f_{G,r} \geq f_{G,1}$ for any $r > 1$.

▶ **Corollary 7** (Monotonicity). *If $0 < r \leq r'$ then, for any digraph G, $f_{G,r} \leq f_{G,r'}$.*

Corollary 7 follows immediately from Theorem 6 since $f_{G,r}(\{v\}) \leq f_{G,r'}(\{v\})$ for all $v \in V(G)$.

The second corollary says that adding more mutants can't decrease the fixation probability and has been assumed in the literature, without proof. However, although it appears obvious at first, it is somewhat subtle: we have seen that adding more mutants can actually decrease the probability of a particular vertex becoming a mutant at the next step of the process.

▶ **Corollary 8** (Subset domination). *For any digraph G and any $r > 0$, if $S \subseteq S' \subseteq V(G)$, then $f_{G,r}(S) \leq f_{G,r}(S')$.*

Note that, although we have introduced the continuous-time version of the process for technical reasons, to draw conclusions about the original, discrete-time Moran process, the continuous-time version may actually be a more realistic model than the discrete-time version.

1.2 Previous Work

Fixation probabilities in the Moran process are known for regular graphs [17] and stars (complete bipartite graphs $K_{1,k}$) [6]. Lieberman et al. [17] give classes of directed graphs with a parameter k, and claim that the fixation probability on these graphs tends to $1 - r^{-k}$ for large graphs. While these graphs do seem to have very large fixation probability, we have shown this specific claim to be incorrect for $k = 5$ [9]. Very recently, it has been claimed [15] that taking $k = \Theta(n^{1/6})$ does yield fixation probability tending to 1 (as a function of n, independently of r) but no rigorous proof is given. Other work has investigated the possibility of so-called "suppressors", graphs having fixation probability less than that given by (1) for at least some range of values for r [19, 20].

There is a more complicated version of the Moran process in which the fitness of a vertex is determined by its expected payoff when playing some two-player game against a randomly chosen neighbour [27, 26]. In this version of the process, mutants play the game with one strategy and non-mutants play the game with another. The ordinary Moran process corresponds to the special case of this game in which the mutant and non-mutant strategies give payoffs r and 1, respectively, regardless of the strategy used by the opponent.

Most previous work on absorption times has been in the game-based version of the process, where the added complexity of the model limits analysis to very simple graphs, such as complete graphs, stars and cycles [1, 5].

2 Preliminaries

When k is a positive integer, $[k]$ denotes $\{1, \ldots, k\}$. We consider the evolution of the Moran process [17] on a strongly connected directed graph (digraph). Consider such a digraph $G = (V, E)$ with $n = |V|$. When the process is run on G, each state is a set of vertices $S \subseteq V$. The vertices in S are said to be "mutants". If $|S| = k$ then the total fitness of the state is given by $W_k = n + (r - 1)k$ — each of the k mutants contributes fitness r to the total fitness and each other vertex contributes fitness 1. Except where stated otherwise, we assume that $r > 1$. The starting state is chosen uniformly at random from the one-element subsets of $V(G)$. From a state S with $|S| = k$, the process evolves as follows. First, a vertex u is chosen to reproduce. The probability that vertex u is chosen is r/W_k if u is a mutant and $1/W_k$ otherwise. Given that u will reproduce, a directed edge (u, v) is chosen uniformly at random from $\{(u, v') \mid (u, v') \in E\}$. The state of vertex u in S is copied to v to give the new state, which is $S \cup \{v\}$ if u is a mutant and $S \setminus \{v\}$, otherwise.

A digraph $G = (V, E)$ is Δ-*regular* if every vertex has in- and out-degree Δ. G is *regular* if it is Δ-regular for some natural number Δ. If the Moran process is run on a strongly connected digraph G, there are exactly two absorbing states — $\emptyset$ and $V(G)$. Once one of these states is reached, the process will stay in it forever. The *absorption time* is the number of steps until such a state is reached.

A digraph $G = (V, E)$ is weakly connected if the underlying undirected graph is connected. Given a subset $S \subseteq V$, let m_S^+ be the number of edges from vertices in S to vertices in $V \setminus S$. Let m_S^- be the number of edges from vertices in $V \setminus S$ to S. Note that every regular digraph that is weakly connected is strongly connected.

We sometimes consider the Moran process on an undirected graph $G = (V, E)$. We view the undirected graph as a digraph in which the set E of edges is symmetric (so $(u, v) \in E$ if and only if $(v, u) \in E$).

All proofs are included in the full version, which is available at [11]. To assist the reader, we have used the numbering from the full version in the extended abstract (so there are gaps in the numbering sequence here).

3 Domination

A useful proof technique is to show that the behaviour of the Moran process is stochastically dominated by that of a related process that is easier to analyse. Similarly, it is useful to compare the behaviour of the Moran process, evolving on a digraph G, with that of another Moran process on the same digraph, where the second process starts with more mutants. Recall that the Moran process can be described as a Markov chain $(Y_t)_{t\geq 1}$ where Y_t is the set $S \subseteq V(G)$ of mutants at the t'th step. It would be natural to attempt to establish a coupling between Moran processes $(Y_t)_{t\geq 1}$ and $(Y'_t)_{t\geq 1}$ such that, if $Y_1 \subseteq Y'_1$, then $Y_t \subseteq Y'_t$ for all $t \geq 1$, but this cannot be done, as shown in Section 1.1.3.

To obtain useful dominations, we will consider a continuous-time version of the Moran process. The domination that we construct for the continuous-time process will allow us to draw conclusions about the original (discrete-time) Moran process. In a digraph $G = (V, E)$ where the set of mutants Y have fitness r, let $r_{v,Y} = r$ if $v \in Y$ and $r_{v,Y} = 1$, otherwise. We define the continuous-time version of the Moran process on a digraph $G = (V, E)$ as follows. Starting in configuration $\widetilde{Y}[t]$ at time t, each vertex v waits for a period of time before reproducing. This period of time is chosen, independently of other vertices, according to an exponential distribution with parameter $r_{v,\widetilde{Y}[t]}$. Therefore, the probability that two vertices reproduce at once is zero. Suppose that the first vertex to reproduce after time t is vertex v, at time $t + \tau$. As in the discrete-time version of the process, an out-neighbour w of v is chosen u.a.r. and the new configuration $\widetilde{Y}[t + \tau]$ is $\widetilde{Y}[t] \cup \{w\}$ if v is a mutant and $\widetilde{Y}[t] \setminus \{w\}$, otherwise.

From the definition of the exponential distribution, it is clear that the probability that a particular vertex v is the next to reproduce, from configuration $\widetilde{Y}[t]$, is $r_{v,\widetilde{Y}[t]}/W_{|\widetilde{Y}[t]|}$. Thus, the Moran process (as generalised by Lieberman et al.) is recovered by taking the sequence of configurations each time a vertex reproduces[1] so the continuous-time process is a faithful simulation of the discrete version.

Using the continuous-time version of the process, we are able to prove the coupling lemma (Lemma 5), Theorem 6 and its corollaries, as stated in Section 1.1.3.

4 Regular Digraphs

This section provides upper and lower bounds on the absorption time of the Moran process on regular digraphs. Clearly, the bounds on expected absorption time of undirected graphs from [10] do not apply to digraphs since [10, Theorem 7] gives a polynomial upper bound but our Theorem 4 shows that process takes exponential time on some strongly connected digraphs. Since we will be discussing both undirected graphs and digraphs in this section,

[1] This is closely related to the jump chain, which is defined to be the discrete-time chain whose successive states are the states $\widetilde{Y}[t]$ for the successive times t immediately after the state changes. Thus, the jump chain is the chain of "active" steps of the discrete-time Moran process (see Section 4.2).

we start by observing as Proposition 9 that [10, Theorem 7] can be improved to give an $O(n^3)$ bound on regular undirected graphs. This is certainly not tight (see below) but it is a natural place to begin.

▶ **Proposition 9.** *The expected absorption time of the Moran process on a connected Δ-regular n-vertex undirected graph is at most $\frac{r}{r-1}n^2\Delta$.*

4.1 Definitions

We will use the following standard Markov chain definitions. For more detail, see, for example, [24]. We use $(X_t)_{t\geq 0}$ to denote a discrete-time Markov chain $\mathcal{M}$ with finite state space Ω and transition matrix P. $T_k = \inf\{t \geq 1 \mid X_t = k\}$ is the first passage time for visiting state k (not counting the initial state X_0). The time spent in state i between visits to state k is

$$\gamma_i^k = \sum_{t=0}^{T_k-1} 1_{X_t=i}, \text{ where } X_0 = k.$$

The chain is *irreducible* if, for every pair of states (i, j) there is some $t \geq 0$ such that $\Pr(X_t = j \mid X_0 = i) > 0$. Since Ω is finite, this implies that the chain is recurrent, which means that, for every state $i \in \Omega$, $\Pr(X_t = i \text{ for infinitely many } t) = 1$. We use the following proposition, which (up to minor notational differences) is the special case of [24, Theorem 1.7.6] for finite state spaces.

▶ **Proposition 10.** *Let $\mathcal{M}$ be an irreducible discrete-time Markov chain with finite state space $\Omega = \{0, \ldots, \omega - 1\}$ and transition matrix P. For $k \in \Omega$, let $\lambda = (\lambda_0, \ldots, \lambda_{\omega-1})$ be a vector of non-negative reals with $\lambda_k = 1$, satisfying $\lambda P = \lambda$. Then for every $j \in \Omega$, $E[\gamma_j^k] = \lambda_j$.*

4.2 Active Steps and Absorption Time

We fix $r > 1$ and study the Moran process on a strongly connected Δ-regular n-vertex digraph $G = (V, E)$ with $n > 1$. An *active step* is one at which the number of mutants changes. As explained in the introduction, the number of mutants after each active step corresponds to a one-dimensional random walk on $\{0, \ldots, n\}$ which starts at state 1, absorbs at 0 and n, and has upwards drift $p = \frac{r}{r+1}$.

To derive the properties that we need, we consider a Markov chain $\mathcal{M}$ with state space $\Omega = \{0, \ldots, n + 1\}$. The non-zero entries of the transition matrix P of $\mathcal{M}$ are $P_{0,n+1} = P_{n,n+1} = P_{n+1,1} = 1$ and, for $1 \leq i \leq n-1$, $P_{i,i+1} = p$ and $P_{i,i-1} = 1-p$. Starting from state 1, the chain simulates the one-dimensional walk discussed above. State $n+1$ is visited immediately after an absorbing state of the random walk is reached. From state $n+1$, the chain goes back to state 1 and repeats the random walk. We establish the following key property of $\mathcal{M}$, which is proved in the full version.

▶ **Lemma 11.** *Let $f = (r^n - r^{n-1})/(r^n - 1)$. Define the vector $\lambda = (\lambda_0, \ldots, \lambda_{n+1})$ as follows.*

$$\lambda_0 = 1 - f, \qquad \lambda_n = f, \qquad \lambda_{n+1} = 1,$$
$$\lambda_j = (1 + r)(1 - f)(r^n - r^j)/(r^n - r), \text{ for } 1 \leq j \leq n - 1.$$

Then, for every $j \in \Omega$, $E[\gamma_j^{n+1}] = \lambda_j$.

It is well known [17], and an easy consequence of Lemma 11, that f is the fixation probability of the Moran process on a regular graph. We use the following corollary; upper bound is because $r^n - r^j \leq r^n - 1$ and the lower bound is because $E[\gamma_j^{n+1}]$ is minimised at $j = n - 1$.

▶ **Corollary 12.** *For all* $j \in \{1, \ldots, n-1\}$, $1 - \frac{1}{r^2} \leq E[\gamma_j^{n+1}] \leq 1 + \frac{1}{r}$.

Now, let the Moran process on a digraph G be $(Y_t)_{t\geq 1}$. For each state S, let $p(S) = \Pr(Y_{t+1} \neq S \mid Y_t = S)$ and let $\mu(S) = \inf\{t \geq 1 \mid Y_{t+1} \neq S, Y_1 = S\}$. The random variable $\mu(S)$ is the number of steps for which the process stays in state S after arriving there. It is geometrically distributed with parameter $p(S)$, so $E[\mu(S)] = 1/p(S)$. Let $\tau_1 = 1$. For $j > 1$, let $\tau_j = \inf\{t > \tau_{j-1} \mid Y_t \neq Y_{t-1}\}$. The values $\tau_2, \tau_3, \ldots$ are the active steps and $(Y_{\tau_j})_{j\geq 1}$ is the Moran process with the repeated states omitted. The time spent in state $Y_{\tau_{j-1}}$ is $\tau_j - \tau_{j-1}$ and the absorption time T_{A} of the Moran process is the sum of these times. If we take $X_0 = n+1$, T_{A} is also $\tau_{T_{n+1}-1}$, where T_{n+1} is the first passage time of state $n+1$ in the process $(X_t)_{t\geq 0}$, above. For convenience, let $\ell = T_{n+1} - 1$.

To derive upper and lower bounds for $E[T_{\mathrm{A}}]$, we break the sum into pieces. For $k \in [n-1]$, let $T_{\mathrm{A},k}$ be the total number of steps spent in states with k mutants before absorption. $T_{\mathrm{A},k} = \sum_{j=2}^{\ell} \Psi_{k,j}$, where $\Psi_{k,j}$ is geometrically distributed with parameter $p(Y_{\tau_{j-1}})$ if $|Y_{\tau_{j-1}}| = k$ and $\Psi_{k,j} = 0$, otherwise. Then T_{A} is distributed as $\sum_{k=1}^{n-1} T_{\mathrm{A},k}$. To derive the bounds, let $p_k^+ = \max\{p(S) \mid |S| = k\}$ and $p_k^- = \min\{p(S) \mid |S| = k\}$. Let $T_{\mathrm{A},k}^+ = \sum_{j=2}^{\ell} \Psi_{k,j}^+$ where $\Psi_{k,j}^+$ is geometrically distributed with parameter p_k^- if $|Y_{\tau_{j-1}}| = k$ and $\Psi_{k,j}^+ = 0$, otherwise. Let $T_{\mathrm{A},k}^- = \sum_{j=2}^{\ell} \Psi_{k,j}^-$ where $\Psi_{k,j}^-$ is geometrically distributed with parameter p_k^+ if $|Y_{\tau_{j-1}}| = k$ and $\Psi_{k,j}^- = 0$, otherwise. Then by stochastic domination for the geometric distribution, $\sum_{k=1}^{n-1} E[T_{\mathrm{A},k}^-] \leq E[T_{\mathrm{A}}] \leq \sum_{k=1}^{n-1} E[T_{\mathrm{A},k}^+]$.

Thus, we have upper and lower bounds for $E[T_{\mathrm{A}}]$ as sums of random numbers (the γ_k^{n+1}) of geometric random variables, each having parameter $1/p_k^+$ or $1/p_k^-$ for some k. Corollary 12 shows that $E[\gamma_k^{n+1}]$ is finite; all the parameters $1/p_k^+$ and $1/p_k^-$ are also finite. Wald's equality and Corollary 12 give us the following theorem.

▶ **Theorem 13.** $\left(1 - \frac{1}{r^2}\right) \sum_{k=1}^{n-1} (1/p_k^+) \leq E[T_A] \leq \left(1 + \frac{1}{r}\right) \sum_{k=1}^{n-1} (1/p_k^-)$.

4.3 Upper and Lower Bounds

Observe that, if $|S| = k$ then $p(S) = \frac{rm_S^+}{W_k\Delta} + \frac{m_S^-}{W_k\Delta}$, so $\frac{1}{p(S)} = \frac{W_k\Delta}{rm_S^+ + m_S^-}$. From this and Theorem 13, we get the following, which allows us to prove Theorem 1.

▶ **Corollary 15.** *The expected absorption time of the Moran process on a strongly connected* Δ*-regular* n*-vertex digraph* G *is at least* $\left(1 - \frac{1}{r^2}\right) W_1 \Delta M$ *and at most* $\left(1 + \frac{1}{r}\right) W_n \Delta M$*, where* M *denotes* $\sum_{k=1}^{n-1} 1/\min\left\{rm_S^+ + m_S^- \mid |S| = k\right\}$.

▶ **Theorem 1.** *The expected absorption time of the Moran process on a strongly connected* Δ*-regular* n*-vertex digraph* G *is at least* $\left(\frac{r-1}{r^2}\right) nH_{n-1}$ *and at most* $n^2\Delta$.

Proof. If $|S| = k$ then we have $rm_S^+ + m_S^- \leq (r+1)k\Delta$, which, with Corollary 15, establishes the lower bound. If a digraph is strongly connected, then $m_S^+, m_S^- \geq 1$ when $1 \leq |S| \leq n-1$ so $rm_S^+ + m_S^- \geq r+1$. This, together with Corollary 15, establishes the upper bound. ◀

Note that the upper bound in Theorem 1 generalises the one given in Proposition 9 to the directed case. The following observations follow from special cases of Corollary 15. All match the corresponding bounds from Theorem 1 up to a constant factor (which may depend on r but not on n).

▶ **Observation 16.** *Let* G *be the undirected clique* K_n *(which is* Δ*-regular with* $\Delta = n-1$*). For* S *of size* k*,* $m_S^+ = m_S^- = k(n-k)$*, so Corollary 15 shows that the absorption time is at*

most

$$n\sum_{k=1}^{n-1}\frac{n-1}{k(n-k)} \leq n\sum_{k=1}^{n-1}\frac{n-k}{k(n-k)} + n\sum_{k=1}^{n-1}\frac{k}{k(n-k)} \leq 2nH_{n-1}.$$

▶ **Observation 17.** *Let G be the undirected cycle C_n (which is Δ-regular with $\Delta = 2$). Since the process starts with a single mutant, it is easy to see that the set of mutant vertices must be connected, if it is non-empty. Therefore, $m_S^+ = m_S^- = 2$ for any non-trivial state S that is reachable from the initial configuration, so the absorption time is at least $\left(1-\frac{1}{r^2}\right)\frac{2n}{r+1}\sum_{k=1}^{n-1}\frac{1}{2} = \Omega(n^2)$.*

▶ **Observation 18.** *Let G be the directed n-vertex cycle (which is Δ-regular with $\Delta = 1$). Again, the mutants remain connected; in this case $m_S^+ = m_S^- = 1$ for any non-trivial reachable S so the absorption time is at least $\left(1-\frac{1}{r^2}\right)\frac{n}{r+1}\sum_{k=1}^{n-1}\frac{1}{2} = \Omega(n^2)$.*

4.4 Better Upper Bounds for Undirected Graphs via Isoperimetric Numbers

For an undirected graph G and $S \subseteq V(G)$, let ∂S be the set of edges between S and $V(G)\setminus S$. Then $m_S^+ = m_S^- = |\partial S|$. The isoperimetric number of G was defined by Buser [7] as

$$i(G) = \min\left\{|\partial S|/|S| \mid S \subseteq V(G), 0 < |S| \leq |V(G)|/2\right\}.$$

The quantity $i(G)$ is a discrete analogue of the Cheeger isoperimetric constant. For graphs with good expansion, **Theorem 3** improves the upper bound from Theorem 1, as discussed in Section 1.1.1.

4.5 Families for Which the Upper Bound is Optimal

For every fixed $\Delta > 2$, we construct an infinite family of connected, Δ-regular undirected graphs for which the upper bound in Theorem 1 is optimal, up to a constant factor (which may depend upon r and Δ but not on n).

To do this, we define the graph H_Δ to be $K_{\Delta-2,\Delta-1}$ with the addition of edges forming a cycle on the side with $\Delta-1$ vertices. Now, let $G_{\ell,\Delta}$ be the Δ-regular graph formed from a cycle $C = x_1 \ldots x_\ell x_1$ and ℓ disjoint copies of H_Δ by adding an undirected edge between x_i and each of the vertices of degree $\Delta-1$ in the i'th copy of H_Δ, for each $i \in [\ell]$. Note that $|V(G_{\ell,\Delta})| = 2\ell(\Delta-1)$. Intuitively, the limiting factor for the propagation of mutants through the graph is that is connected only through the cycle C and absorption time on cycles is quadratic (Observation 17; with some work and careful use of stochastic dominations, this can be formalised into a proof of the following theorem.

▶ **Theorem 19.** *For $r > 1$ and sufficiently large ℓ (with respect to r), the expected absorption time of the Moran process on $G_{\ell,\Delta}$ exceeds $\frac{1}{2r}(1-\frac{1}{r})\ell^2$.*

Thus, the $O(n^2)$ upper bound of Theorem 1 is tight up to a constant factor (which may depend on r and Δ, but not on n). For given r and Δ, let $\mathcal{G}$ be the class of all graphs $G_{\ell,\Delta}$ with ℓ large enough for Theorem 19 to apply: this proves **Theorem 2**.

5 General Digraphs

Fix $r > 1$ and let $\varepsilon_r = \min(r-1, 1)$. Theorem 7 of [10] shows that the expected absorption time of the Moran process on a connected n-vertex undirected graph is at most $\frac{r}{r-1}n^4$.

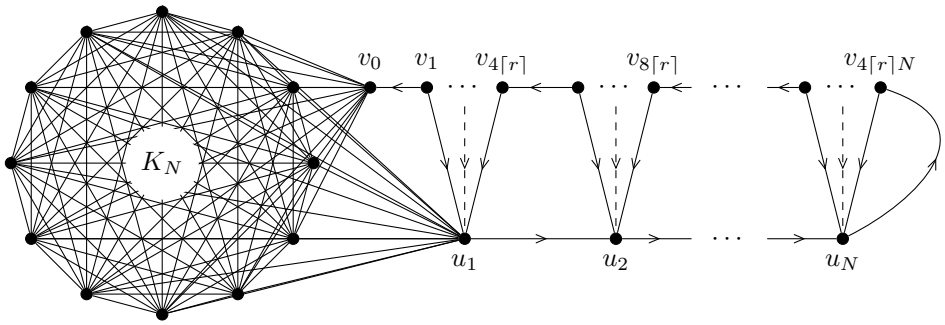


Figure 1 The graph $G_{r,N}$. The edges within the clique are bidirectional; v_0 sends a directed edge to every vertex in the clique and u_1 receives one from each. Other edges are directed as indicated.

Theorem 1 shows that the expected absorption time on a strongly connected Δ-regular digraph is at most $n^2\Delta$. In contrast, we exhibit an infinite family of strongly connected digraphs such that the expected absorption time on an n-vertex graph from the family is $2^{\Omega(n)}$.

Let $G_{r,N}$ be the disjoint union of the complete graph K_N (with bidirectional edges), a directed path $P = u_1 \dots u_N$ and a directed path $Q = v_{4\lceil r\rceil N} \dots v_0$, along with the directed edge $(u_N, v_{4\lceil r\rceil N})$ and the directed edges (x, u_1) and (v_0, x) for every $x \in K_N$ and $(v_{4\lceil r\rceil(i-1)+j}, u_i)$ for each $i \in [N]$, $j \in [4\lceil r\rceil]$ (see Figure 1).

Intuitively, the Moran process on $G_{r,N}$ behaves as follows. With probability close to $\frac{1}{4\lceil r\rceil+2}$, the initial mutant is in the clique. Since there are $N(N-1)$ directed edges within the clique and only $2N$ between it and the rest of the graph, it is reasonable to expect that the behaviour of the mutants within the clique is close to the behaviour of mutants in a complete graph, which is a one-dimensional random walk. The main challenge is to show that, conditioned on the initial mutant being in the clique, enough reproductions happen in the clique in the first N^3 steps of the process that the probability of the clique becoming at least half full of mutants within the first N^3 steps of the process is at least $\varepsilon_r/8$. From this state, we show that both fixation and extinction are likely to be slow.

Fixation cannot occur unless all vertices in P, the first part of the path, become mutants. It is relatively straightforward to show that, conditioned on the initial mutant being in the clique, the probability of P filling with mutants within the first $T^*(N)$ steps is also at most $\varepsilon_r/32$. The argument proceeds by a stochastic domination. We allow only the mutant in P that is farthest from the clique to be replaced by a non-mutant, so the mutants in P occupy an initial segment of the path. The number of mutants in P behaves as a random walk with downward drift, since the "mutant frontier" in P is more likely to be pushed back by the edges from Q than it is to move forwards.

On the other hand, extinction cannot happen while there are still mutants in the clique. As observed in Section 1.1.1, the number of mutants in a complete graph is a random walk with upward drift r. We may approximate the behaviour of the mutants in the clique by a one-dimensional random walk with upward drift r that has a non-absorbing barrier at N. (Since the clique is not the whole graph, filling the clique with mutants does not correspond to absorption of the process.) Such a walk is expected to take exponentially long to return to zero. This allows us to show that, conditioned on the initial mutant being in the clique and the clique reaching $N/2$ mutants by step N^3, the probability that the number of mutants in the clique reaches zero by step $T^*(N)$ is at most $\varepsilon_r/32$, where $T^*(N) = \lfloor \varepsilon_r(2^N - 1)/32 \rfloor$.

Therefore, the conditional probability of extinction within the first $T^*(N)$ steps is also at most $\varepsilon_r/32$.

Combining these results, and using Markov's inequality, we obtain the lower bound on the expected absorption time of the Moran process on $G_{r,N}$, given by **Theorem 4** of Section 1.1.2.

References

1 Tibor Antal and István Scheuring. Fixation of strategies for an evolutionary game in finite populations. *Bulletin of Mathematical Biology*, 68(8):1923–1944, 2006.

2 Chalee Asavathiratham, Sandip Roy, Bernard Lesieutre, and George Verghese. The influence model. *IEEE Control Systems*, 21(6):52–64, 2001.

3 Eli Berge. Dynamic monopolies of constant size. *Journal of Combinatorial Theory, Series B*, 83(2):191–200, 2001.

4 Béla Bollobás. The isoperimetric number of random regular graphs. *European J. Combin.*, 9(3):241–244, 1988.

5 M. Broom, C. Hadjicrysanthou, and J. Rychtář. Evolutionary games on graphs and the speed of the evolutionary process. *Proceedings of the Royal Society A*, 466(2117):1327–1346, 2010.

6 M. Broom and J. Rychtář. An analysis of the fixation probability of a mutant on special classes of non-directed graphs. *Proceedings of the Royal Society A*, 464(2098):2609–2627, 2008.

7 Peter Buser. Cubic graphs and the first eigenvalue of a Riemann surface. *Math. Z.*, 162(1):87–99, 1978.

8 M. Cemil Azizoğlu and Ömer Eğecioğlu. The bisection width and the isoperimetric number of arrays. *Discrete Appl. Math.*, 138(1–2):3–12, 2004.

9 Josep Díaz, Leslie Ann Goldberg, George B. Mertzios, David Richerby, Maria J. Serna, and Paul G. Spirakis. On the fixation probability of superstars. *Proceedings of the Royal Society A*, 469(2156):20130193, 2013.

10 Josep Díaz, Leslie Ann Goldberg, George B. Mertzios, David Richerby, Maria J. Serna, and Paul G. Spirakis. Approximating fixation probabilities in the generalized Moran process. *Algorithmica*, to appear.

11 Josep Díaz, Leslie Ann Goldberg, David Richerby, and Maria J. Serna. Absorption time of the moran process. *CoRR*, abs/1311.7631, 2013.

12 Herbert Gintis. *Game Theory Evolving: A Problem-Centered Introduction to Modeling Strategic Interaction*. Princeton University Press, 2000.

13 G. R. Grimmett and D. R. Stirzaker. *Probability and Random Processes*. Oxford University Press, 3rd edition, 2001.

14 B. Houchmandzadeh and M. Vallade. The fixation probability of a beneficial mutation in a geographically structured population. *New Journal of Physics*, 13:073020, 2011.

15 A. Jamieson-Lane and C. Hauert. Fixation probabilities on superstars, revisited and revised. ArXiv 1312.6333, 2013.

16 David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proc. 9th ACM International Conference on Knowledge Discovery and Data Mining*, pages 137–146. ACM, 2003.

17 Erez Lieberman, Christoph Hauert, and Martin A. Nowak. Evolutionary dynamics on graphs. *Nature*, 433(7023):312–316, 2005.

18 Thomas M. Liggett. *Stochastic Interacting Systems: Contact, Voter and Exclusion Processes*. Springer, 1999.

19 George B. Mertzios, Sotiris E. Nikoletseas, Christoforos Raptopoulos, and Paul G. Spirakis. Natural models for evolution on networks. *Theoretical Computer Science*, 477:76–95, 2013.

20 George B. Mertzios and Paul G. Spirakis. Strong bounds for evolution in networks. In *Proc. 40th International Colloquium on Automata, Languages and Programming (ICALP 2013)*, volume 7966 of *LNCS*, pages 657–668. Springer, 2013.

21 Bojan Mohar. Isoperimetric numbers of graphs. *Journal of Combinatorial Theory, Series B*, 47(3):274–291, 1989.

22 P. A. P. Moran. Random processes in genetics. *Proceedings of the Cambridge Philosophical Society*, 54(1):60–71, 1958.

23 Pu-Yan Nie and Pei-Ai Zhang. Fixation time for evolutionary graphs. *International Journal of Modern Physics B*, 24(27):5285–5293, 2010.

24 James R. Norris. *Markov Chains*. Cambridge University Press, 1998.

25 Paulo Shakarian and Patrick Roos. Fast and deterministic computation of fixation probability in evolutionary graphs. In *Proc. 6th International Conference on Computational Intelligence and Bioinformatics*, pages 753–012. ACTA Press, 2011.

26 Paulo Shakarian, Patrick Roos, and Anthony Johnson. A review of evolutionary graph theory with applications to game theory. *Biosystems*, 107:66–80, 2012.

27 Christine Taylor, Drew Fudenberg, Akira Sasaki, and Martin A. Nowak. Evolutionary game dynamics in finite populations. *Bulletin of Mathematical Biology*, 66(6):1621–1644, 2004.

Sampling a Uniform Solution of a Quadratic Equation Modulo a Prime Power*

Chandan Dubey and Thomas Holenstein

Institute for Theoretical Computer Science, ETH Zürich
{chandan.dubey,thomas.holenstein}@inf.ethz.ch

Abstract

An n-ary integral quadratic form is a formal expression $Q(x_1, \cdots, x_n) = \sum_{1 \le i,j \le n} a_{ij} x_i x_j$ in n variables $x_1, \cdots, x_n$, where $a_{ij} = a_{ji} \in \mathbb{Z}$. We present a randomized polynomial time algorithm that given a quadratic form $Q(x_1, \cdots, x_n)$, a prime p, a positive integer k and an integer t, samples a uniform solution of $Q(x_1, \cdots, x_n) \equiv t \bmod p^k$.

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems

Keywords and phrases Quadratic Forms, Lattices, Modular, p-adic

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.643

1 Introduction

Let R be a commutative ring with unity and $\mathrm{R}^\times$ be the set of units (i.e., invertible elements) of R. A quadratic form over the ring R in n-formal variables $x_1, \cdots, x_n$ in an expression $\sum_{1 \le i,j \le n} a_{ij} x_i x_j$, where $a_{ij} = a_{ji} \in \mathrm{R}$. A quadratic form can equivalently be represented by a symmetric matrix $\mathtt{Q}^n = (a_{ij})$ such that $Q(x_1, \cdots, x_n) = (x_1, \cdots, x_n)'\mathtt{Q}(x_1, \cdots, x_n)$. The quadratic form is called integral if $\mathrm{R} = \mathbb{Z}$ and the determinant of the quadratic form Q is defined as $\det(\mathtt{Q})$.

Quadratic forms are central to various branches of Mathematics, including number theory, linear algebra, group theory, and Lie theory. They also appear in several areas of Computer Science like Cryptography and Lattices. Several modern factorization algorithms, including Dixon's algorithm [6], the continued fractions method, and the quadratic sieve; try to solve $x^2 \equiv t \bmod n$, where n is the number being factorized. They also arise naturally as the ℓ_2 norm of lattice vectors.

It is not surprising that the study of quadratic forms predates Gauss, who gave the law of quadratic reciprocity and contributed a great deal in the study of quadratic forms, including a complete classification of binary quadratic forms (i.e., $n = 2$). Another giant leap was made by Minkowski in his "Geometry of Numbers" [11], which proposed a geometric method to solve problems in number theory. Minkowski also gave explicit formulae to calculate the number of solutions $\mathbf{x} = (x_1, \cdots, x_n) \in (\mathbb{Z}/p^k\mathbb{Z})^n$ to the equation $\mathbf{x}'\mathtt{Q}\mathbf{x} \equiv t \bmod p^k$. Several alternatives are available for counting. We refer to [17, 12, 19, 10, 5, 7, 9], and note that many of these papers (also) solve much more general problems. As an example, [17] gives an ingenious Gaussian sum technique to count solutions in case p does not divide $2t \det(\mathtt{Q})$.

The case of the prime $p = 2$ is tricky and needs careful analysis. Pall [13] pointed out that the work of Minkowski omits many details, resulting in errors for the case of prime 2. Later, Watson [18] found errors in the fixes suggested by Pall. It is believed by the community that the work by Watson does not contain any errors.

* This research was supported by the Swiss National Science Foundation, grant no. 200021–132508.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 643–653

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

We remark that typically mathematicians are mainly interested in counting the number of solutions if k is "large enough".[1] One reason for this is that once k is large enough, increasing k by 1 simply multiplies the number of solutions by p^{n-1}. Another reason is that the corresponding normalized quantity (the local density, which is the number of solutions divided by $p^{k(n-1)}$ for k large enough) seems to be the "mathematically natural quantity". It arises in many places, for example in (some forms of) the celebrated Siegel mass formula [17].

On the question of finding any solution (in contrast to sampling one uniformly at random), we are aware of two relevant results. The first [1, 15] solves $x^2 - ky^2 \equiv m \pmod{n}$ for composite n, when the factorization of n is unknown. The second and more relevant is the work done by Hartung [8]. For odd p, he gives a correct polynomial time algorithm to find one solution of $Q \equiv t \bmod p^k$ (thought it seems to be safe to say that the possibility of this was folklore before). Unfortunately, his construction seems to contain errors for the case $p = 2$ (e. g., he divides by 2 in the proof of the relevant Lemma 3.3.1 pp. 47–48).

Our Contribution

Apart from the difficulty of giving correct formulae for $p = 2$, the method of Minkowski (and others, including the Gaussian sum method) for counting the number of solutions of $\mathbf{x}'\mathtt{Q}\mathbf{x} \equiv t \bmod p^k$ has another drawback. It is not constructive in the sense that it does not provide a way to sample uniform solutions to the equation. In this work, we give an alternate way of counting solutions, and thus by the above remarks, and alternate way to compute the local density. Our algorithm also yields a Las Vegas algorithm that, given an integral quadratic form $\mathtt{Q}$, a prime p, a positive integer k and an integer $t \in \mathbb{Z}/p^k\mathbb{Z}$, runs in time $\mathrm{poly}(n, k, \log p)$ and samples a uniform random solution of $\mathbf{x}'\mathtt{Q}\mathbf{x} \equiv t \bmod p^k$.

2 Preliminaries

Integers and ring elements are denoted by lowercase letters, vectors by bold lowercase letters and matrices by typewriter uppercase letters. The i'th component of a vector $\mathbf{v}$ is denoted by v_i. We use the notation $(v_1, \cdots, v_n)$ for a column vector and the transpose of matrix $\mathtt{A}$ is denoted by $\mathtt{A}'$. The matrix $\mathtt{A}^n$ will denote a $n \times n$ square matrix. If $\mathtt{Q}_1^n, \mathtt{Q}_2^m$ are matrices, then the *direct product* of $\mathtt{Q}_1$ and $\mathtt{Q}_2$ is denoted by $\mathtt{Q}_1 \oplus \mathtt{Q}_2$ and is defined as $\mathrm{diag}(\mathtt{Q}_1, \mathtt{Q}_2) = \begin{pmatrix} \mathtt{Q}_1 & 0 \\ 0 & \mathtt{Q}_2 \end{pmatrix}$. Given two matrices $\mathtt{Q}_1$ and $\mathtt{Q}_2$ with the same number of rows, $[\mathtt{Q}_1, \mathtt{Q}_2]$ is the matrix which is obtained by concatenating the two matrices columnwise. A matrix is called unimodular if it is an integer $n \times n$ matrix with determinant 1.

Let R be a commutative ring with unity and $\mathrm{R}^\times$ be the set of units (i.e., invertible elements) of R. If $\mathtt{Q} \in \mathrm{R}^{n \times n}$ is a square matrix, the *adjugate* of $\mathtt{Q}$ is defined as the transpose of the cofactor matrix and is denoted by $\mathrm{adj}(\mathtt{Q})$. The matrix $\mathtt{Q}$ is invertible if and only if $\det(\mathtt{Q})$ is a unit of the R. In this case, $\mathrm{adj}(\mathtt{Q}) = \det(\mathtt{Q})\mathtt{Q}^{-1}$. The set of invertible $n \times n$ matrices over R is denoted by $\mathrm{GL}_n(\mathrm{R})$. The subset of matrices with determinant 1 will be denoted by $\mathrm{SL}_n(\mathrm{R})$. For every prime p and positive integer k, we define the ring $\mathbb{Z}/p^k\mathbb{Z} = \{0, \cdots, p^k - 1\}$, where product and addition is defined modulo p^k.

▶ **Fact 1.** *A matrix* $\mathtt{U}$ *is in* $\mathrm{GL}_n(\mathrm{R})$ *iff* $\det(\mathtt{U}) \in \mathrm{R}^\times$.

[1] If k is 1 larger than the largest power of p in $8 \cdot t \cdot \det(\mathtt{Q})$, the following holds.

For quadratic forms, the prime 2 is special and all primes except 2 will be called *odd primes*. Let p be a prime, and a be an integer. Then, $\mathrm{ord}_p(a)$ is the highest power of p such that $p^{\mathrm{ord}_p(a)}$ divides a. We let $\mathrm{ord}_p(0) = \infty$. The p-coprime part of a is then $\textsc{copr}_p(a) = \frac{a}{p^{\mathrm{ord}_p(a)}}$. Note that $\textsc{copr}_p(a)$ is by definition a unit of $\mathbb{Z}/p\mathbb{Z}$. For a positive integer q, one writes $a \equiv b \bmod q$, if q divides $a - b$. By $x := a \bmod q$, we mean that x is assigned the unique value $b \in \mathbb{Z}/q\mathbb{Z}$ such that $b \equiv a \bmod q$. An integer t is called a *quadratic residue* modulo q if $\gcd(t, q) = 1$ and $x^2 \equiv t \bmod q$ has a solution.

▶ **Definition 2.** Let p be an odd prime, and t be a positive integer. Then, the Legendre-symbol of t with respect to p is defined as follows.

$$\left(\frac{t}{p}\right) := t^{(p-1)/2} \bmod p = \begin{cases} 1 & \text{if } t \text{ is a quadratic residue modulo } p, \text{ and } t \not\equiv 0 \pmod p \\ 0 & \text{if } t \equiv 0 \pmod p, \\ -1 & \text{otherwise.} \end{cases}$$

For the prime 2, there is an extension of Legendre symbol called the Kronecker symbol. It is defined for odd integers t and $\left(\frac{t}{2}\right)$ equals 1 iff $t \equiv \pm 1 \bmod 8$, -1 if $t \equiv \pm 3 \bmod 8$, and 0 otherwise. The p-sign of t, denoted $\mathrm{sgn}_p(t)$, is defined as $\left(\frac{\textsc{copr}_p(t)}{p}\right)$ for odd primes p and $\textsc{copr}_2(t) \bmod 8$ otherwise.

▶ **Fact 3.** *Let p be an odd prime. Then, there are $\frac{p-1}{2}$ quadratic residues and $\frac{p-1}{2}$ quadratic non-residues modulo p.*

An integer t is a square modulo q if there exists an integer x such that $x^2 \equiv t \pmod q$. The integer x is called the *square root* of t modulo q. If no such x exists, then t is a non-square modulo q.

▶ **Definition 4.** Let q be a prime power. A vector $\mathbf{v} \in (\mathbb{Z}/q\mathbb{Z})^n$ is called primitive if there exists a component v_i, $i \in [n]$, of $\mathbf{v}$ such that $\gcd(v_i, q) = 1$. Otherwise, the vector $\mathbf{v}$ is non-primitive.

▶ **Definition 5.** Let p be a prime, k be a positive integer and x be an element of $\mathbb{Z}/p^k\mathbb{Z}$. The p-expansion of x is x written in base p i.e., $x = d_0(x) + d_1(x) \cdot p + \cdots + d_{k-1}(x) \cdot p^{k-1}$, where $d_i(x) \in \mathbb{Z}/p\mathbb{Z}$ for $i \in \{0, \cdots, k-1\}$, is called the i'th *digit* of x.

For two sets A and B, the symbol $A \leftrightarrow B$ means that there is bijection between A and B.

Quadratic Form

An n-ary quadratic form over a ring R is a symmetric matrix $\mathtt{Q} \in \mathrm{R}^{n \times n}$, interpreted as the following polynomial in n formal variables $x_1, \cdots, x_n$ of uniform degree 2.

$$\sum_{1 \le i,j \le n} \mathtt{Q}_{ij} x_i x_j = \mathtt{Q}_{11} x_1^2 + \mathtt{Q}_{12} x_1 x_2 + \cdots = \mathbf{x}' \mathtt{Q} \mathbf{x}$$

The quadratic form is called *integral* if it is defined over the ring $\mathbb{Z}$ and is also positive definite if for all non-zero column vector $\mathbf{x}$, $\mathbf{x}'\mathtt{Q}\mathbf{x} > 0$. This work deals with integral quadratic forms, henceforth called simply *quadratic forms*. The *determinant* of the quadratic form is defined as $\det(\mathtt{Q})$. A quadratic form is called *diagonal* if $\mathtt{Q}$ is a diagonal matrix.

Given a set of formal variables $\mathbf{x} = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix}'$ one can make a linear change of variables to $\mathbf{y} = \begin{pmatrix} y_1 & \cdots & y_n \end{pmatrix}'$ using a matrix $\mathtt{U} \in \mathrm{R}^{n \times n}$ by setting $\mathbf{y} = \mathtt{U}\mathbf{x}$. If additionally, $\mathtt{U}$ is invertible over R i.e., $\mathtt{U} \in \mathrm{GL_n(R)}$, then this change of variables is reversible over the ring. We now define the equivalence of quadratic forms over the ring R (compare with Lattice Isomorphism).

▶ **Definition 6.** Let $\mathtt{Q}_1^n, \mathtt{Q}_2^n$ be quadratic forms over a ring R. They are called R-*equivalent* if there exists a $\mathtt{U} \in \mathrm{GL_n(R)}$ such that $\mathtt{Q}_2 = \mathtt{U}'\mathtt{Q}_1\mathtt{U}$.

If $\mathrm{R} = \mathbb{Z}/q\mathbb{Z}$, for some positive integer q, then two integral quadratic forms $\mathtt{Q}_1^n$ and $\mathtt{Q}_2^n$ will be called q-equivalent (denoted, $\mathtt{Q}_1 \overset{q}{\sim} \mathtt{Q}_2$) if there exists a matrix $\mathtt{U} \in \mathrm{GL_n}(\mathbb{Z}/q\mathbb{Z})$ such that $\mathtt{Q}_2 \equiv \mathtt{U}'\mathtt{Q}_1\mathtt{U} \pmod q$.

Let $\mathtt{Q}^n$ be a n-ary integral quadratic form, and q, t be positive integers. If the equation $\mathbf{x}'\mathtt{Q}\mathbf{x} \equiv t \pmod q$ has a solution then we say that t has a q-representation in $\mathtt{Q}$ (or t has a representation in $\mathtt{Q}$ over $\mathbb{Z}/q\mathbb{Z}$). Solutions $\mathbf{x} \in (\mathbb{Z}/q\mathbb{Z})^n$ to the equation are called q-*representations* of t in $\mathtt{Q}$. We classify the representations into two categories: *primitive* and *non-primitive*, see definition 4. The set of non-primitive, primitive and all p^k-representations of t in $\mathtt{Q}$ is denoted by $\mathrm{A}_{\mathrm{p^k}}(\mathtt{Q}, t), \mathrm{B}_{\mathrm{p^k}}(\mathtt{Q}, t)$ and $\mathrm{C}_{\mathrm{p^k}}(\mathtt{Q}, t)$. Their respective sizes are denoted by $\mathfrak{A}_{\mathrm{p^k}}(\mathtt{Q}, t), \mathfrak{B}_{\mathrm{p^k}}(\mathtt{Q}, t)$ and $\mathfrak{C}_{\mathrm{p^k}}(\mathtt{Q}, t)$ respectively.

Randomized Algorithms

Our randomized algorithms are Las Vegas algorithms. They either fail and output nothing, or produce a correct answer. The probability of failure is bounded by a constant. Thus, for any $\delta > 0$, it is possible to repeat the algorithm $O(\log \frac{1}{\delta})$ times and succeed with probability at least $1 - \delta$. Henceforth, these algorithms will be called *randomized algorithms*.

Througout this paper, we will say that an algorithm runs in polynomial time if it runs in time $\mathrm{poly}(n, k, \log(p))$.

3 Technical Overview

Given a quadratic form over a ring R, one can classify them according to the following equivalence. Two quadratic forms are equivalent over R if one can be obtained from the other by an invertible linear change of variables over R. For example, x^2 and $2y^2$ are equivalent over the field of reals $\mathbb{R}$ because the transformations $x \to \sqrt{2}y$ and $y \to \frac{1}{\sqrt{2}}x$ are inverse of each other in $\mathbb{R}$, are linear and transform x^2 to $2y^2$ and $2y^2$ to x^2 respectively. Thus, over $\mathbb{R}$ instead of trying to solve both x^2 and $2y^2$ separately, one can instead solve x^2 and then use the invertible linear transformation to map the solutions of x^2 to the solutions of $2y^2$. It is well known that every quadratic form in n-variables over $\mathbb{R}$ is equivalent to $\sum_{i=1}^{a} x_i^2 - \sum_{i=a+1}^{n} x_i^2$, for some $a \in [n]$. This is known as the Sylvester's Law of inertia. The following lemma shows that for counting/finding solutions over a ring R, it suffices to do it for an equivalent quadratic form.

▶ **Lemma 7.** *Let p be a prime, k, t be positive integers, $\mathtt{Q}$ be an integral quadratic form, $\mathtt{U} \in \mathrm{GL_n}(\mathbb{Z}/p^k\mathbb{Z})$ and $\mathtt{S} = \mathtt{U}'\mathtt{Q}\mathtt{U} \bmod p^k$. Then, $\mathrm{A}_{\mathrm{p^k}}(\mathtt{Q}, t) \leftrightarrow \mathrm{A}_{\mathrm{p^k}}(\mathtt{S}, t)$, and $\mathrm{B}_{\mathrm{p^k}}(\mathtt{Q}, t) \leftrightarrow \mathrm{B}_{\mathrm{p^k}}(\mathtt{S}, t)$.*

Proof. The map $\mathbf{x} \to \mathtt{U}\mathbf{x}$ preserves the primitiveness of the vector $\mathbf{x} \in (\mathbb{Z}/p^k\mathbb{Z})^n$ and is bijective because $\mathtt{U}$ is an invertible matrix over $\mathbb{Z}/p^k\mathbb{Z}$. The lemma follows from the equality $(\mathtt{U}\mathbf{x})'\mathtt{Q}(\mathtt{U}\mathbf{x}) \equiv \mathbf{x}'\mathtt{S}\mathbf{x} \bmod p^k$. ◀

For the $\mathrm{R} = \mathbb{Z}/p^k\mathbb{Z}$ such that p is odd, there always exists an equivalent quadratic form which is also diagonal (see [5], Theorem 2, page 369). Additionally, one can explicitly find the invertible change of variables that turns it into a diagonal quadratic form. The situation is tricky over the ring $\mathbb{Z}/2^k\mathbb{Z}$. Here, it might not be possible to eliminate all mixed terms, i.e., terms of the form $a_{ij}x_ix_j$ with $i \neq j$. For example, consider the quadratic form xy over

$\mathbb{Z}/2^k\mathbb{Z}$, for some positive k. An invertible linear change of variables over $\mathbb{Z}/2^k\mathbb{Z}$ is of the following form.

$$\begin{array}{ll} x \to a_1x_1 + a_2x_2 & \begin{pmatrix} a_1 & a_2 \\ b_1 & b_2 \end{pmatrix} \text{ invertible over } \mathbb{Z}/2^k\mathbb{Z} \\ y \to b_1x_1 + b_2x_2 & \end{array}$$

The mixed term after this transformation is $a_1b_2 + a_2b_1$. As $a_1b_2 + a_2b_1 \bmod 2$ is the same as the determinant of the change of variables above i.e., $a_1b_2 - a_2b_1$ modulo 2; it is not possible for a transformation in $\mathrm{GL}_2(\mathbb{Z}/2^k\mathbb{Z})$ to eliminate the mixed term. Instead, one can show that over $\mathbb{Z}/2^k\mathbb{Z}$ it is possible to get an equivalent form where the mixed terms are disjoint i.e., both x_ix_j and x_ix_k do not appear, where $i \neq j \neq k$. One captures this form by the following definition.

▶ **Definition 8.** A matrix $\mathtt{D}^n$ over integers is in a block diagonal form if it is a direct sum of type I and type II forms; where type I form is an integer while type II is a matrix of the form $\begin{pmatrix} 2^{\ell+1}a & 2^\ell b \\ 2^\ell b & 2^{\ell+1}c \end{pmatrix}$ with b odd.

The following theorem is folklore and is also implicit in the proof of Theorem 2 on page 369 in [5].

▶ **Theorem 9.** *Let $\mathtt{Q}^n$ be an integral quadratic form, p be a prime, and k be a positive integer. Then, there is an algorithm that takes time $O(n^4k\log p)$ and produces a matrix $\mathtt{U} \in \mathrm{SL}_n(\mathbb{Z}/p^k\mathbb{Z})$ such that $\mathtt{U}'\mathtt{Q}\mathtt{U} \pmod{p^k}$, is a diagonal matrix for odd primes p and a block diagonal matrix (in the sense of Definition 8) for $p = 2$.*

The next simplification is achieved by the following Lemma.

▶ **Lemma 10.** *Let $\mathtt{Q}^n$ be a quadratic form, p be a prime, k be a positive integer and t, s be integers such that $\mathrm{ord}_p(t \bmod p^k) = \mathrm{ord}_p(s \bmod p^k)$ and $\mathrm{sgn}_p(s \bmod p^k) = \mathrm{sgn}_p(t \bmod p^k)$. Then, $\mathrm{A}_{\mathrm{p}^k}(\mathtt{Q}, t) \leftrightarrow \mathrm{A}_{\mathrm{p}^k}(\mathtt{Q}, s)$, and $\mathrm{B}_{\mathrm{p}^k}(\mathtt{Q}, t) \leftrightarrow \mathrm{B}_{\mathrm{p}^k}(\mathtt{Q}, s)$.*

The pair $(\mathrm{ord}_p(t \bmod p^k), \mathrm{sgn}_p(t \bmod p^k))$ is called the p^k-symbol of t and is denoted by $\mathrm{sym}_{p^k}(t)$. By Lemma 10, the count depends only on the p^k-symbol of t. For notational convenience, we define the following sets.

$$\textsc{ord} = \{\infty, 0, \cdots, k-1\} \quad \textsc{sgn} = \begin{cases} \{1, -1\} & p \text{ is an odd prime} \\ \{1, 3, 5, 7\} & \text{otherwise} \end{cases} \tag{1}$$

Note that, there are p^k different possibilities for t over $\mathbb{Z}/p^k\mathbb{Z}$ but only $(2k+1)$ possibilities for $\mathrm{sym}_{p^k}(t)$ for odd primes and $(4k+1)$ for 2 (the extra 1 is for 0). The following definition is useful in reducing the problem of counting representations in higher dimensions to the problem of counting representations for individual blocks in a block diagonal form.

▶ **Definition 11.** Let p be a prime, k be a positive integer, $t \in \mathbb{Z}/p^k\mathbb{Z}$ be an integer, and γ_1, γ_2 be symbols. Then, the (γ_1, γ_2)-split size of t over $\mathbb{Z}/p^k\mathbb{Z}$, denoted $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p}^k}(\gamma_1, \gamma_2)$, is the size of the following set,

$$\mathrm{S}^{\mathrm{t}}_{\mathrm{p}^k}(\gamma_1, \gamma_2) = \left\{(a, b) \in (\mathbb{Z}/p^k\mathbb{Z})^2 \mid \mathrm{sym}_{p^k}(a) = \gamma_1, \mathrm{sym}_{p^k}(b) = \gamma_2, t \equiv a + b \bmod p^k\right\}$$

If $\mathfrak{A}_{\mathrm{p}^k}(\mathtt{Q}, \gamma)$ is defined as $\mathfrak{A}_{\mathrm{p}^k}(\mathtt{Q}, a)$ for any $a \in \{x \in \mathbb{Z}/p^k\mathbb{Z} \mid \mathrm{sym}_{p^k}(x) = \gamma\}$, and $\mathfrak{B}_{\mathrm{p}^k}(\mathtt{Q}, \gamma), \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}, \gamma)$ are defined similarly, then the following Lemma gives us a way to reduce the problem of counting solutions from $\mathtt{D} = \mathtt{D}_1 \oplus \mathtt{D}_2$ to counting solutions for $\mathtt{D}_1$ and $\mathtt{D}_2$.

▶ **Lemma 12.** *Let* $\mathtt{Q} = \mathrm{diag}(\mathtt{Q}_1, \mathtt{Q}_2)$ *be an integral quadratic form,* p *be a prime,* k *be a positive integer and* $t \in \mathbb{Z}/p^k\mathbb{Z}$*. Then,*

$$\mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}, t) = \sum_{\gamma_1,\gamma_2 \in \mathrm{ORD} \times \mathrm{SGN}} \mathfrak{S}^{\mathrm{t}}_{\mathrm{p}^k}(\gamma_1, \gamma_2) \cdot \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}_1, \gamma_1) \cdot \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}_2, \gamma_2)$$

$$\mathfrak{A}_{\mathrm{p}^k}(\mathtt{Q}, t) = \sum_{\gamma_1,\gamma_2 \in \mathrm{ORD} \times \mathrm{SGN}} \mathfrak{S}^{\mathrm{t}}_{\mathrm{p}^k}(\gamma_1, \gamma_2) \cdot \mathfrak{A}_{\mathrm{p}^k}(\mathtt{Q}_1, \gamma_1) \cdot \mathfrak{A}_{\mathrm{p}^k}(\mathtt{Q}_2, \gamma_2)$$

Proof. The formula for the total number of representations of t by $\mathtt{Q}$ over $\mathbb{Z}/p^k\mathbb{Z}$ follows from the calculations below. The same calculation works for the number of non-primitive representations because an representation of t by $\mathtt{Q}$ is non-primitive iff every component of the representation is non-primitive.

$$\begin{aligned}
\mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}, t) &= \sum_{a \in \mathbb{Z}/p^k\mathbb{Z}} \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}_1, a) \cdot \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}_2, t-a) \\
&= \sum_{a \in \mathbb{Z}/p^k\mathbb{Z}} \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}_1, \mathrm{sym}_{p^k}(a)) \cdot \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}_2, \mathrm{sym}_{p^k}(t-a)) \\
&= \sum_{\gamma_1,\gamma_2 \in \mathrm{ORD} \times \mathrm{SGN}} \mathfrak{S}^{\mathrm{t}}_{\mathrm{p}^k}(\gamma_1, \gamma_2) \cdot \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}_1, \gamma_1) \cdot \mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}_2, \gamma_2)
\end{aligned}$$

◀

Overview of the Algorithm

Given $(\mathtt{Q}^n, p, k, t)$ our counting algorithm for finding $\mathfrak{C}_{\mathrm{p}^k}(\mathtt{Q}, t)$ is as follows.

1. Block diagonalize $\mathtt{Q}$ over $\mathbb{Z}/p^k\mathbb{Z}$ using Theorem 9. Let $\mathtt{D}^n = \mathtt{D}_1 \oplus \cdots \oplus \mathtt{D}_m$ be the block diagonal form returned by the algorithm. Recall, each $\mathtt{D}_i$ is either Type I i.e., an integer, or Type II (only when $p = 2$).
2. For each symbol $\gamma \in \mathrm{ORD} \times \mathrm{SGN}$ and $i \in [m]$, calculate $\mathfrak{C}_{\mathrm{p}^k}(\mathtt{D}_i, \gamma)$. The case of prime 2 is handled separately and needs careful analysis for Type II blocks.
3. For each triple $\gamma, \gamma_1, \gamma_2 \in \mathrm{ORD} \times \mathrm{SGN}$ compute the size of split classes i.e., $\mathcal{S}^{\gamma}_{p^k}(\gamma_1, \gamma_2)$
4. Compute $\mathfrak{C}_{\mathrm{p}^k}(\mathtt{D}_1 \oplus \cdots \oplus \mathtt{D}_i, \gamma)$ for each $\gamma \in \mathrm{ORD} \times \mathrm{SGN}$ and $i \in [m]$, using Lemma 12.
5. Output $\mathfrak{C}_{\mathrm{p}^k}(\mathtt{D}, \mathrm{sym}_{p^k}(t))$.

Because of the remarks in the introduction in the paper, this algorithm can also be used to compute what mathematicians call the "local density". We defer the details to the full paper.

Furthermore, this algorithm can be generalized to sample uniform representations. However, also here we defer the description of the details of this to the full paper. Nevertheless, the following two theorems are the main contribution of this paper.

▶ **Theorem 13.** *Let* $\mathtt{Q}^n$ *be an integral quadratic form,* k *be a positive integer, and* t *be an element of* $\mathbb{Z}/2^k\mathbb{Z}$*. Then, there exists a polynomial time algorithm that samples a uniform (primitive/non-primitive) representation of* t *by* $\mathtt{Q}$ *over* $\mathbb{Z}/2^k\mathbb{Z}$*.*

In other words, the algorithm is able to output a uniform representation, a representation which is uniform among the primitive ones, and a representation which is uniform among the non-primitive ones.

▶ **Theorem 14.** *Let* $\mathtt{Q}^n$ *be an integral quadratic form,* p *be an odd prime,* k *be a positive integer,* t *be an element of* $\mathbb{Z}/p^k\mathbb{Z}$*. Then, there is a polynomial time algorithm that fails and outputs a special symbol* $\perp$ *with probability at most* $\frac{1}{3}$*. Otherwise, the algorithm outputs a uniform (primitive/non-primitive)* p^k*-representation of* t *by* $\mathtt{Q}$*.*

Obviously the algorithm in this theorem can be repeated $\log(1/\delta)$ times to make the error probability at most δ.

4 Counting Representations: A Brief Overview

In order to explain the main ideas of the paper, we sketch in more detail how we count the number of representations.

4.1 Counting for $n = 1$

Counting both the primitive and non-primitive solutions of $Qx^2 = t \bmod p^k$ is rather simple, and of course well known. Ignoring some corner cases (such as $t = 0 \bmod p^k$), we can see that writing $x = x_0 p^\alpha$ with $\gcd(x_0, p) = 1$ we need $x_0^2 Q p^{2\alpha} = t \bmod p^k$, so that we certainly need that $\operatorname{ord}_p Q \geq \operatorname{ord}_p(t)$ and $\operatorname{ord}_p Q - \operatorname{ord}_p t$ is even. Furthermore, in case p is odd the Legendre-symbols of Q and t need to be the same, and it is not hard to show that these are the exact conditions. In case $p = 2$, of course $\textsc{copr}_2(Q) = \textsc{copr}_2(t)$ is required.

4.2 Counting for Type II matrics

Recall Definition 8 of a type II quadratic form. In this section, we solve the representation problem for Type II matrices over $\mathbb{Z}/2^k\mathbb{Z}$. But first we define a scaled version of a type II matrix.

▶ **Definition 15.** A two-by-two matrix of the following form is called type II* matrix.

$$\begin{pmatrix} a & b/2 \\ b/2 & c \end{pmatrix} \qquad a, b, c \in \mathbb{Z}, b \text{ odd}$$

Additionally, in this section we will think of type II* as the following quadratic form in formal variables x_1, x_2 which take values in the ring $\mathbb{Z}/2^k\mathbb{Z}$.

$$ax_1^2 + bx_1x_2 + cx_2^2 \qquad a, b, c \in \mathbb{Z}, b \text{ odd} . \tag{2}$$

In order to count the number of representations, the following lemma is key.

▶ **Lemma 16.** *Let $\mathtt{Q}^* = (a, b, c)$, b odd be a type II* integral quadratic form, and t, k be positive integers. If $a_1, a_2 \in \mathbb{Z}/2\mathbb{Z}$ be such that (a_1, a_2) represent t over $\mathbb{Z}/2\mathbb{Z}$ and either a_1 or a_2 is odd then there are exactly 2^{k-1} distinct representations (x_1, x_2) of t over $\mathbb{Z}/2^k\mathbb{Z}$ such that $x_1 \equiv a_1 \pmod 2, x_2 \equiv a_2 \pmod 2$.*

Proof. We prove this by induction on k. We show that given an representation y_1, y_2 of t over the ring $\mathbb{Z}/2^i\mathbb{Z}$, for $i \geq 1$, such that at least one of y_1, y_2 is odd there are exactly two representations z_1, z_2 of t over the ring $\mathbb{Z}/2^{i+1}\mathbb{Z}$ such that $z_1 \equiv y_1 \pmod{2^i}, z_2 \equiv y_2 \pmod{2^i}$.

Let (y_1, y_2) be an representation of t by $\mathtt{Q}^*$ over $\mathbb{Z}/2^i\mathbb{Z}$. Then, the pair of integers (z_1, z_2) such that $(z_1, z_2) \equiv (y_1, y_2) \pmod{2^i}$ is an representation of t over $\mathbb{Z}/2^{i+1}\mathbb{Z}$ iff

$$\begin{array}{ll} z_1 \equiv y_1 + b_1 \cdot 2^i \pmod{2^{i+1}} & z_2 \equiv y_2 + b_2 \cdot 2^i \pmod{2^{i+1}} \\ b_1, b_2 \in \{0, 1\} & az_1^2 + bz_1z_2 + cz_2^2 \equiv t \pmod{2^{i+1}} \end{array} \tag{3}$$

Plugging in the values of z_1 and z_2 and re-arranging we get the following equation.

$$(bb_2y_1 + bb_1y_2)2^i \equiv t - (ay_1^2 + by_1y_2 + cy_2^2) \pmod{2^{i+1}} \tag{4}$$

As b is odd, b is invertible over $\mathbb{Z}/2^{i+1}\mathbb{Z}$. By assumption, y_1, y_2 represent t over $\mathbb{Z}/2^i\mathbb{Z}$ and hence 2^i divides $t - (ay_1^2 + by_1y_2 + cy_2^2)$. The equation 4 reduces to the following equation.

$$b_2y_1 + b_1y_2 \equiv \frac{t - (ay_1^2 + by_1y_2 + cy_2^2)}{2^i b} \pmod 2 \tag{5}$$

We now split the proof in two cases: i) when y_1 is odd, and ii) when y_1 is even and y_2 is odd.

y_1 odd. For each choice of $b_1 \in \{0,1\}$ there is a unique choice for b_2 because $y_1 \equiv 1 \pmod 2$.

$$b_1 \in \{0,1\} \qquad b_2 = \frac{t - (ay_1^2 + by_1y_2 + cy_2^2)}{2^i b} - b_1y_2 \pmod 2$$

y_1 even. In this case, $y_2 \equiv 1 \pmod 2$ and so b_2 can be chosen freely.

$$b_2 \in \{0,1\} \qquad b_1 = \frac{t - (ay_1^2 + by_1y_2 + cy_2^2)}{2^i b} \pmod 2$$

◀

Using this lemma, in order to count the number of representations of t modulo 2^k by a type II* matrix, we simply first check how many of the three pairs $(0,1)$, $(1,0)$, and $(1,1)$ represent t modulo 2. The remaining case (where both a_1 and a_2 are even) obviously requires that t is divisible by 4, and can be settled by a simple recursion (where one represents $t/4$ modulo 2^{k-2}). We defer a detailed description to the full version of the paper.

4.3 Calculating the Split Classes

Our next step is to calculate the split size i.e., $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2)$.

Let p be a prime, k be a positive integer and $t \in \mathbb{Z}/p^k\mathbb{Z}$. In this section, we calculate the value $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2)$ for all possible symbol pairs (γ_1, γ_2) over $\mathbb{Z}/p^k\mathbb{Z}$. We also show that $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2)$ only depends on the p^k-symbol of t and can also be written as $\mathfrak{S}^{\gamma}_{\mathrm{p^k}}(\gamma_1, \gamma_2)$, where $\gamma = \mathrm{sym}_{p^k}(t)$.

For a p^k-symbol γ, suppose $\mathrm{S}_{\mathrm{p^k}}(\gamma) = \{x \in \mathbb{Z}/p^k\mathbb{Z} \mid \mathrm{sym}_{p^k}(x) = \gamma\}$ and $\mathfrak{S}_{\mathrm{p^k}}(\gamma)$ be the cardinality of $\mathrm{S}_{\mathrm{p^k}}(\gamma)$. Then, the following lemma calculates, for each $a \in \mathbb{Z}/p^k\mathbb{Z}$, the number of elements in $\mathbb{Z}/p^k\mathbb{Z}$ with the same p^k-symbol as a.

▶ **Lemma 17.** *Let p be a prime, k be a positive integer and $a \in \mathbb{Z}/p^k\mathbb{Z}$ be a non-zero integer. Then,*

$$\mathfrak{S}_{\mathrm{p^k}}(\mathrm{sym}_{p^k}(a)) = \begin{cases} \max\{2^{k-\mathrm{ord}_2(a)-3}, 1\} & \text{if } p = 2 \\ \frac{p-1}{2}p^{k-\mathrm{ord}_p(a)-1} & \text{otherwise.} \end{cases}$$

Proof. Let $x \in \mathbb{Z}/p^k\mathbb{Z}$ be an element with the same p-symbol as a. Then, $\mathrm{ord}_p(x) = \mathrm{ord}_p(a)$ and $\mathrm{sgn}_p(x) = \mathrm{sgn}_p(t)$. Recall the p-expansion of x i.e., definition 5. There are k digits in the p-expansion of x for $x \in \mathbb{Z}/p^k\mathbb{Z}$; first $\mathrm{ord}_p(a)$ of which must be identically 0.

For odd prime p, $\mathrm{sgn}_p(x) = \mathrm{sgn}_p(a)$ iff $\left(\frac{\mathrm{COPR}_p(x)\,\mathrm{COPR}_p(t)}{p}\right) = 1$. Thus, the $(\mathrm{ord}_p(a)+1)$'th digit of x must be a non-zero element of $\mathbb{Z}/p\mathbb{Z}$ with the same sign as $\left(\frac{\mathrm{COPR}_p(a)}{p}\right)$. By Fact 3, there are $\frac{p-1}{2}$ possibilities for the $(\mathrm{ord}_p(a)+1)$'th digit of x. The rest can be chosen freely from $\mathbb{Z}/p\mathbb{Z}$.

For the prime 2, $\mathrm{sgn}_2(x) = \mathrm{sgn}_2(a)$ iff $\mathrm{COPR}_2(x) \equiv \mathrm{COPR}_2(a) \bmod 8$. Thus, the digits $(\mathrm{ord}_p(a)+1), \cdots, (\mathrm{ord}_p(a)+2)$ of x must match those of a. The rest can be chosen freely from $\mathbb{Z}/2\mathbb{Z}$. ◀

The following two lemmas show that if $\mathrm{ord}_p(t) \neq \mathrm{ord}_p(a)$ then the symbol of $t-a \bmod p^k$ is the same for every element of $\mathbb{Z}/p^k\mathbb{Z}$ which has the same p^k-symbol as a.

▶ **Lemma 18.** *Let k be a positive integer and a, t be elements of the ring $\mathbb{Z}/2^k\mathbb{Z}$. Then, the 2^k-symbol of $t-a$ can be computed from $\mathrm{sym}_{2^k}(t)$ and $\mathrm{sym}_{2^k}(a)$.*

Proof. The 2^k-symbol of $s = (t-a) \bmod 2^k$ can be calculated as follows.

$$\mathrm{ord}_2(s) = \min\{\mathrm{ord}_2(t), \mathrm{ord}_2(a)\}$$

$$\mathrm{COPR}_2(t-a) = \begin{cases} 2^{\mathrm{ord}_2(t)-\mathrm{ord}_2(a)}\,\mathrm{COPR}_2(t) - \mathrm{COPR}_2(a) & \text{if } \mathrm{ord}_2(t) > \mathrm{ord}_2(a) \\ \mathrm{COPR}_2(t) - 2^{\mathrm{ord}_2(a)-\mathrm{ord}_2(t)}\,\mathrm{COPR}_2(a) & \text{otherwise.} \end{cases}$$

$$\mathrm{COPR}_2(s) = \begin{cases} \mathrm{COPR}_2(t-a) \bmod 2^{k-\mathrm{ord}_2(a)} & \text{if } \mathrm{ord}_2(t) > \mathrm{ord}_2(a) \\ \mathrm{COPR}_2(t-a) \bmod 2^{k-\mathrm{ord}_2(t)} & \text{otherwise.} \end{cases}$$

The quantity $\mathrm{COPR}_2(s) \bmod 8$ can be computed from $\mathrm{COPR}_2(t) \bmod 8, \mathrm{COPR}_2(a) \bmod 8$, $\mathrm{ord}_2(t)$ and $\mathrm{ord}_2(a)$. ◀

▶ **Lemma 19.** *Let p be an odd prime, k be a positive integer and a, t be elements of the ring $\mathbb{Z}/p^k\mathbb{Z}$ such that $\mathrm{ord}_p(t) \neq \mathrm{ord}_p(a)$. Then, the p^k-symbol of $t-a$ can be computed from $\mathrm{sym}_{p^k}(t)$ and $\mathrm{sym}_{p^k}(a)$.*

Proof. The p^k-symbol of $t-a$ can be calculated as follows.

$$\mathrm{ord}_p(t-a) = \min\{\mathrm{ord}_p(t), \mathrm{ord}_p(a)\}$$

$$\mathrm{COPR}_p(t-a) = \begin{cases} p^{\mathrm{ord}_p(t)-\mathrm{ord}_p(a)}\,\mathrm{COPR}_p(t) - \mathrm{COPR}_p(a) & \text{if } \mathrm{ord}_p(t) > \mathrm{ord}_p(a) \\ \mathrm{COPR}_p(t) - p^{\mathrm{ord}_p(a)-\mathrm{ord}_p(t)}\,\mathrm{COPR}_p(a) & \text{otherwise.} \end{cases}$$

$$\left(\frac{\mathrm{COPR}_p(t-a)}{p}\right) = \begin{cases} \left(\frac{-\mathrm{COPR}_p(a)}{p}\right) & \text{if } \mathrm{ord}_p(t) > \mathrm{ord}_p(a) \\ \left(\frac{\mathrm{COPR}_p(t)}{p}\right) & \text{otherwise.} \end{cases}$$

◀

The next lemma is from [14].

▶ **Lemma 20.** *For an odd prime p, and non-zero $a \in \mathbb{Z}/p\mathbb{Z}$ the number of tuples $(x, x+a) \in (\mathbb{Z}/p\mathbb{Z})^2$ such that $\left(\frac{x}{p}\right) = s_1$, $\left(\frac{x+a}{p}\right) = s_2$ and $s_1, s_2 \in \{-1, 1\}$ is given by the following formula.*

$$\frac{1}{4} \cdot \left\{ p - (p \bmod 4) - \left(\frac{-1}{p}\right) \cdot \left(1 + s_1 \left(\frac{a}{p}\right)\right) \cdot \left(1 + s_2 \left(\frac{-a}{p}\right)\right) \right\} \tag{6}$$

The following lemma gives the size of the $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2)$ for all possible p^k-symbol pairs over the ring $\mathbb{Z}/p^k\mathbb{Z}$.

▶ **Lemma 21.** *Let $t \in \mathbb{Z}/p^k\mathbb{Z}$, p be a prime, and k be a positive integer. Then, the size of the $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2)$ for all possible p-symbol pairs over the ring $\mathbb{Z}/p^k\mathbb{Z}$ can be computed as follows.*

1. *if $\mathrm{ord}_p(\gamma_1), \mathrm{ord}_p(\gamma_2) > \mathrm{ord}_p(t)$ then $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2) = 0$.*
2. *if $\mathrm{ord}_p(\gamma_1) \neq \mathrm{ord}_p(t)$ then $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2) = \mathfrak{S}_{\mathrm{p^k}}(\gamma_1)$, for exactly one γ_2 and is 0 otherwise.*
3. *if $\mathrm{ord}_p(\gamma_2) \neq \mathrm{ord}_p(t)$ then $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2) = \mathfrak{S}_{\mathrm{p^k}}(\gamma_2)$, for exactly one γ_1 and is 0 otherwise.*
4. *if $\mathrm{ord}_p(\gamma_2) = \mathrm{ord}_p(\gamma_1) = \mathrm{ord}_p(t)$ then $\mathfrak{S}^{\mathrm{t}}_{\mathrm{p^k}}(\gamma_1, \gamma_2)$ is 0 for $p = 2$ and otherwise it is calculated by substituting $\left(\frac{a}{p}\right) = \mathrm{sgn}_p(\gamma_1)$, $s_1 = \mathrm{sgn}_p(\gamma_2)$ and $s_2 = \left(\frac{\mathrm{COPR}_p(t)}{p}\right)$ in equation 6 and multiplying the result by $p^{k-\mathrm{ord}_p(t)-1}$.*

Proof. The $\mathfrak{S}^{t}_{p^k}(\gamma_1, \gamma_2)$ is defined as follows.

$$\mathfrak{S}^{t}_{p^k}(\gamma_1, \gamma_2) = \left|\{(a, b) \in (\mathbb{Z}/p^k\mathbb{Z})^2 \mid \mathrm{sym}_{p^k}(a) = \gamma_1, \mathrm{sym}_{p^k}(b) = \gamma_2, \text{ and, } t \equiv a + b \bmod p^k\}\right|$$

If both $\mathrm{ord}_p(\gamma_1)$ and $\mathrm{ord}_p(\gamma_2)$ are larger than $\mathrm{ord}_p(t)$ then it is not possible for such a pair to add up to t modulo p^k. If $\mathrm{ord}_p(\gamma_1)$ or $\mathrm{ord}_p(\gamma_2)$ is different from $\mathrm{ord}_p(t)$ then the correctness follows from lemma 18, when $p = 2$ and lemma 19 otherwise. Otherwise, $\mathrm{ord}_p(t) = \mathrm{ord}_p(\gamma_1) = \mathrm{ord}_p(\gamma_2)$. This is not possible in case $p = 2$ because the sum of two numbers of the same 2-order is always a number of higher 2-order. For odd prime p, we are looking for number of solutions in $\mathbb{Z}/p^k\mathbb{Z}$ of the following equation.

$$\begin{aligned} & p^{\mathrm{ord}_p(t)}\,\mathrm{COPR}_p(a) + p^{\mathrm{ord}_p(t)}\,\mathrm{COPR}_p(b) \equiv p^{\mathrm{ord}_p(t)}\,\mathrm{COPR}_p(t) \bmod p^k \\ & \iff \mathrm{COPR}_p(a) + \mathrm{COPR}_p(b) \equiv \mathrm{COPR}_p(t) \bmod p^{k-\mathrm{ord}_p(t)} \end{aligned} \tag{7}$$

The number of solutions of equation 4.3 modulo p is given by Lemma 20. The other $(k - \mathrm{ord}_p(t) - 1)$ digits in the p-expansion of $\mathrm{COPR}_p(a)$ can be chosen freely. Thus, the number of possibilities multiply by $p^{k-\mathrm{ord}_p(t)-1}$. ◀

▶ **Lemma 22.** *Let p be a prime, k be a positive integer, $t \in \mathbb{Z}/p^k\mathbb{Z}$ and γ_1, γ_2 be two one dimensional symbols. Then, $\mathfrak{S}^{t}_{p^k}(\gamma_1, \gamma_2)$ only depends on the p-symbol of $t \bmod p^k$.*

Proof. The calculation of $\mathfrak{S}^{t}_{p^k}(\gamma_1, \gamma_2)$ in Lemma 21 only depends on $\mathrm{ord}_p(t \bmod p^k)$ and $\mathrm{sgn}_p(t \bmod p^k)$. ◀

Thus, we mean the same thing by $\mathfrak{S}^{t}_{p^k}(\gamma_1, \gamma_2)$ and $\mathfrak{S}_{p^k}{}^{\mathrm{sym}_{p^k}(t)}(\gamma_1, \gamma_2)$.

Acknowledgements. We thank the reviewers for plenty of extremely helpful comments. In particular, we would like the two reviewers who gave extremely detailed comments and suggestion for improvements. We will incorporate more of their suggestions in the full version. All omissions and remaining errors are the authors responsability.

References

1 Leonard M. Adleman, Dennis R. Estes, and Kevin S. McCurley. Solving bivariate quadratic congruences in random polynomial time. *Mathematics of Computation*, 48(177):17–28, 1987.

2 Zenon Ivanovich Borevich and Igor Rostislavovich Shafarevich. *Number theory*, volume 20. Academic Press, 1986.

3 Sungmun Cho. Group schemes and local densities of quadratic lattices in residue characteristic 2. *arXiv:1210.7625v2*, preprint.

4 John Conway and Neil J. A. Sloane. Low-dimensional lattices IV. The mass formula. *Proc. R. Soc. Lond. A*, 419:259–286, 1988.

5 John Conway and Neil J. A. Sloane. *Sphere packings, lattices and groups*, volume 290. Springer, 1999.

6 John D Dixon. Asymptotically fast factorization of integers. *Mathematics of computation*, 36(153):255–260, 1981.

7 Wee Teck Gan and Jiu-Kang Yu. Group schemes and local densities. *Duke mathematical journal*, 105(3), 497–524, 2000.

8 Rupert Hartung. *Computational problems of quadratic forms: complexity and cryptographic perspectives*. Ph. D. thesis, Goethe-Universität Frankfurt a. M., 2008, http://publikationen.ub.uni-frankfurt.de/volltexte/2008/5444/pdf/HartungRupert.pdf, 2008.

9 Jonathan Hanke. Local densities and explicit bounds for representability by a quadratic form. *Duke mathematical journal*, 124(2), 351–388, 2004.

10 Yoshiyuki Kitaoka. *Arithmetic of quadratic forms*, volume 106. Cambridge University Press, 1999.

11 Hermann Minkowski. *Geometrie der Zahlen*. Berlin, 1910.

12 Onorato Timothy O'Meara. *Introduction to quadratic forms*, volume 117. Springer, 1973.

13 Gordon Pall. The weight of a genus of positive n-ary quadratic forms. In *Proc. Sympos. Pure Math*, volume 8, pages 95–105, 1965.

14 Oskar Perron. Bemerkungen über die Verteilung der quadratischen Reste. *Mathematische Zeitschrift*, 56:122–130, 1952.

15 John M. Pollard, Claus-Peter Schnorr An efficient solution of the congruence $x^2 + ky^2 = m \pmod{n}$. *IEEE Transactions on Information Theory* 33(5):702–709.

16 Victor Shoup. *A computational introduction to number theory and algebra*. Cambridge University Press, 2009.

17 Carl Ludwig Siegel. Über die analytische Theorie der quadratischen Formen. *The Annals of Mathematics*, 36(3):527–606, 1935.

18 G. L. Watson. The 2-adic density of a quadratic form. *Mathematika*, 23(01):94–106, 1976.

19 Tonghai Yang. An Explicit Formula for Local Densities of Quadratic Forms. *Journal of Number Theory*, 72:309–256, 1998.

Unidirectional Input/Output Streaming Complexity of Reversal and Sorting*

Nathanaël François[1], Rahul Jain[2], and Frédéric Magniez[3]

1 **Univ Paris Diderot, Sorbonne Paris-Cité, LIAFA, CNRS, 75205 Paris, France**
nathanael.francois@liafa.univ-paris-diderot.fr
2 **CQT and CS Department, National University of Singapore**
rahul@comp.nus.edu.sg
3 **CNRS, LIAFA, Univ Paris Diderot, Sorbonne Paris-Cité, 75205 Paris, France**
frederic.magniez@cnrs.fr

Abstract

We consider unidirectional data streams with restricted access, such as read-only and write-only streams. For read-write streams, we also introduce a new complexity measure called expansion, the ratio between the space used on the stream and the input size.

We give tight bounds for the complexity of reversing a stream of length n in several of the possible models. In the read-only and write-only model, we show that p-pass algorithms need memory space $\Theta(n/p)$. But if either the output stream or the input stream is read-write, then the complexity falls to $\Theta(n/p^2)$. It becomes polylog(n) if $p = \mathrm{O}(\log n)$ and both streams are read-write.

We also study the complexity of sorting a stream and give two algorithms with small expansion. Our main sorting algorithm is randomized and has $\mathrm{O}(1)$ expansion, $\mathrm{O}(\log n)$ passes and $\mathrm{O}(\log n)$ memory.

1998 ACM Subject Classification F.2.0 Analysis of Algorithms and Problem Complexity

Keywords and phrases Streaming Algorithms, Multiple Streams, Reversal, Sorting

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.654

1 Introduction

Background and Motivations. Streaming algorithms have been studied for estimating statistics, checking properties and computing functions (more often with sublinear outputs) on massive inputs for several years. However, less is known on computing functions that also have a massive output, and therefore require two data streams, one for the input and one for the output. The notion of reversal complexity on a multi-tape machine, which can be related to streaming complexity with multiple streams, was first introduced in 1970 by Kameda and Vollmar [12] for decision problems. The model however was not explored again until 1991 when Chen and Yap [5] considered the computable functions in this model, and gave an algorithm for sorting using two streams with $\mathrm{O}(\log n)$ passes and (internal) memory space of size $\mathrm{O}(\log n)$. This was a significant improvement over the lower bound for the case of a single stream which requires $\Omega(n/s)$ passes, where s is the size of the memory space, as proved by Munro and Paterson [15].

* Partially supported by the French ANR Blanc project ANR-12-BS02-005 (RDAM), the Singapore Ministry of Education Tier 3 Grant and also the Core Grants of the Centre for Quantum Technologies, Singapore.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 654–668

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

Recently the interest for complexity in models with multiple streams has been renewed. Indeed, streams are now considered as a model of external storage allowing multiple sequential passes on them. Hernich and Schweikardt [10] gave reductions from classical complexity classes to the classes of problems decidable with O(1) streams. Grohe, Koch, Hernich and Scwheikardt [9, 8] also gave several tight lower bounds for multi-stream algorithms using $o(\log n)$ passes. Gagie [7] showed that two streams achieve perfect compression in polylogarithmic memory space and a polylogarithmic number of passes. Beame, Huynh, Jayram and Rudra [3, 2] also proved several lower bounds on approximating frequency moments with multiple streams and $o(\log n)$ passes. However, there is a distinct lack of lower bounds for multi-stream algorithms with $\Omega(\log n)$ passes. Indeed, most classical tools (such as reduction to communication complexity) for proving lower bounds on streaming algorithms fail for multiple streams. One exception is Ruhl's [17] W-Stream and StreamSort models. In W-Stream each (unidirectional) stream is alternatively read-only and write-only, and passes are synchronized; while this model seems similar to ours, it is actually much less powerful and most lower bounds match the naive linear algorithms. StreamSort is the W-Stream model augmented with a sorting oracle and yields more interesting results, but of course trivializes the problem we study in this paper. Indeed, most classical tools (such as reduction to communication complexity) for proving lower bounds on streaming algorithms fail for multiple streams.

We therefore take an opposite approach and restrict the number of streams to two, the *input stream* and the *output stream*. This leads to our notion of *input/output streaming algorithms*. For many scenarios, this model is in fact more realistic since, as an external storage, it comes at a cost to use several streams simultaneously. In fact, the idea of using only input and output and constrained secondary storage can also be found in other models, such as [1]. Additionally, we only allow passes in one direction on both streams. This may looks too restrictive since bidirectional passes provide exponential speedup in several decision problems, as exhibited by several works [14, 11, 4, 13, 6]. However, changing the direction of processing a stream can have a higher cost: most hard drives spin in only one direction and reading them in the other is not as practical. We also consider three types of stream accesses: *read-only* (for the input stream), *write-only* (for the output stream) and *read-write* (for any or both streams). We emphasise that by write-only stream we mean that once the algorithm has written in a cell, it can never overwrite it.

For read-write streams, we introduce the new complexity measure of *expansion* of a streaming algorithm, i. e. the ratio between the maximal size of the stream during the computation and the size of the input (this concept does not make sense for read-only and write-only stream, which cannot be used for intermediate computations). While space on external storage is by definition not as constrained as memory space, in the case of massive inputs requiring large hard drives, expanding them even by a factor $O(\log n)$ is not reasonable. To the best of our knowledge, while it has sometimes been implicitely limited to 1 or O(1), this measure of complexity has never been studied before for potential trade-offs with space or time.

We then study two problems, Reverse and Sort, in the model of input/output streaming algorithms. Both problems consists in copying the content of the input stream to the output stream, in reverse order for the first problem, and in sorted order for the second one. Because of its importance in real applications, Sort has been extensively studied in many contexts. However, there is still a lot to say about this problem in face of new models of computation that massive data arise. Reverse can be seen as the simplest instance of Sort. It is natural to study its complexity since we forbid any reverse pass on streams. Moreover, any algorithm

for Reverse gives a way to implement each of the efficient bidirectional (and single stream) algorithms of [14, 13, 6] in our model, thus giving another example of a speed-up with multiple streams.

Our Results. In Section 3, we give tight bounds for the complexity of Reverse. We provide deterministic algorithms that are optimal even against randomized ones. Using communication complexity to prove lower bounds with multiple streams is inherently difficult as soon as there are multiple passes on both. Indeed, it is possible to copy one stream on the other one, and since heads move separately, the algorithm can access at any part of the second stream while processing the first one. In the read-only/write-only model, we instead skip communication complexity and prove directly with information theory that a p-pass algorithm needs space $\Omega(n/p)$ (Theorem 3). In the read-only/read-write model, we similarly show that any p-pass algorithm needs space $\Omega(n/p^2)$ (Theorem 10). We provide an algorithm achieving that bound by copying large (unreversed) blocks from the input, and then placing correctly one element from each of the blocks during each pass (Theorem 8). Similar results hold in the read-write/write-only model (Theorems 9 and 12). Last, we consider the read-write/read-write model and give a $\mathrm{O}(\log n)$-pass algorithm with polylogarithmic memory space (Theorem 13). All our algorithms presented make extensive use of the ability to only write part of a stream during a pass. In the more restrictive W-Stream model where this is not possible, as shown by [17], a p-pass algorithm requires memory $\Omega(n/p)$.

In Section 4, we consider problem Sort. Even with two streams only, it appears to have a tight bound, as the algorithm by Chen and Yap [5] matches the lower bound proved by Grohe, Hernich and Scwheikardt [8]. However, Chen and Yap's algorithm was not designed with expansion in mind: it works by replicating the input n times, thus using $\mathrm{O}(n^2)$ cells on the input and output streams. In this context, linear expansion of the input is not reasonable. We therefore give two algorithms with two streams that improve on it. The first one is deterministic. Similarly to the former algorithm, it is based on Merge Sort, but has expansion $\mathrm{O}(\log n)$ instead of $\Omega(n)$. It also uses $\mathrm{O}(\log n)$ passes and space $\mathrm{O}(\log n)$ (Theorem 14). The second one is randomized and based on Quick Sort. It has expansion $\mathrm{O}(1)$, $\mathrm{O}(\log n)$ passes and memory $\mathrm{O}(\log n)$ (Theorem 15).

2 Preliminaries

In streaming algorithms (see [16] for an introduction), a *pass* on a string $w \in \Sigma^n$, for some finite alphabet Σ, means that w is given as a *stream* $w[1], w[2], \ldots, w[n]$, which arrives sequentially, i. e., letter by letter in this order. Depending on the model, w may or may not be modified. For simplicity, we always assume $|\Sigma| = \mathrm{O}(n)$, except when explicitly stated. Otherwise, one should transpose our algorithms such that the letters are also read sequentially, say bit by bit. We now fix in the rest of the paper such a finite alphabet Σ equipped with a total order.

We consider *input/output streaming algorithms* with two data streams X and Y. Stream X initially contains the input, and stream Y is initially empty and will contain the output at the end of the execution of the algorithm. We denote by $X[i]$ the i-th cell of stream X, and similarly for Y. The size of a stream is the number of its cells containing some data. Our algorithms have also access to a random access memory M, usually of sublinear size in the input size.

We then parameterize such algorithms by the operations allowed on the input stream X (read-only or read-write), on the output stream Y (write-only or read-write), the number of

passes, the bit-size of the memory M and the expansion of the streams with regard to the size of the input. Whenever we mention the memory of an algorithm, we mean a random-access memory.

▶ **Definition 1** (Input/Output streaming algorithm)**.** Let I be either RO or RW, and let O be either WO or RW. Then a *$p(n)$-pass I/O streaming algorithm* with space $s(n)$ is an algorithm that, given $w \in \Sigma^n$ as a stream X, produces its output on an initially empty stream Y and such that:

- It performs $p(n)$ sequential passes in total on X and Y;
- It maintains a memory M of size at most $s(n)$ bits while processing X, Y;
- If I is RO, X cannot be modified;
- If O is WO, Y cannot be read and each cell of Y can be written only once.

Moreover, the algorithm has expansion $\lambda(n)$ if streams X, Y have length (in number of cells) at most $\lambda(n) \times n$ during its execution, for all input $w \in \Sigma^n$.

Observe that we do not mention any running time in our definition. Indeed, all our lower bounds will be stated independently of it. Moreover, all our algorithms process each letter from each stream in polylogarithmic time. They also have also polylogarithmic preprocessing and postprocessing times.

A usual way to get an algorithm with expansion > 1 consists in annotating streams with a larger alphabet Σ', while keeping (artificially) the same number of annotated cells. In that case, one can simulate one annotated cell using $(\log|\Sigma'|)/\log|\Sigma|$ non-annotated cells.

For simplicity, we assume further that the input length n is always given to the algorithm in advance. Nonetheless, all our algorithms can be adapted to the case in which n is unknown until the end of a pass. Moreover, this assumption only makes our lower bounds stronger.

We will use the two usual notions of randomized computing.

Monte Carlo: An algorithm computes a function f on Σ^n *with error* $\varepsilon \leq 1/3$ if for all inputs $w \in \Sigma^n$, it outputs $f(x)$ with probability at least $1 - \varepsilon$.

Las Vegas: An algorithm computes a function f on Σ^n *with failure* $\varepsilon \leq 1/2$ if for all inputs $w \in \Sigma^n$, it outputs $f(x)$ with probability at least $1 - \varepsilon$, otherwise it gives no output.

The two functions we study in this paper in term of streaming complexity are now formally defined.

▶ **Definition 2** (Reverse and Sort)**.** For $w = w[1]w[2]\dots w[n] \in \Sigma^n$, let us define Reverse$(w)$ as $w[n]w[n-1]\dots w[1]$. When Σ has a total order, also define Sort(w) as the sorted permutation of w.

For simplicity when proving lower bounds, for the proofs of Theorems 3, 10 and 12, instead of considering algorithms for Reverse writing $Reverse(X)$ on Y with passes from left to right, we will have them write X on Y with passes from right to left, which is equivalent with a relabeling of Y. The algorithms presented for upper bounds, however, read better in the model where we write Reverse(X) on Y using a pass from left tor right.

When we describe a streaming algorithm using pseudo-code, a 'For' loop will always correspond to a single pass on each stream, except when explicitly mentioned otherwise. Most algorithms will consist of a 'While' loop including a constant number of 'For' loops, i. e. passes. Therefore the number of iterations of the 'While' loop will give us the number of passes.

Throughout the paper, we always give randomized lower bounds. While communication complexity arguments fail in the context of mutliple streams, we use information theory arguments. They have recently been proved to be a powerful tool for proving lower bounds

in communication complexity, and therefore for streaming algorithms. Here we not only use them directly, but also there is no direct interpretation in term of communication complexity.

Let us remind now the notions of entropy H and mutual information I. Let X, Y, Z be three random variables. Then :

$$\begin{aligned} \mathrm{H}(X) &= -\,\mathbb{E}_{x\leftarrow X} \log \Pr(X = x), \\ \mathrm{H}(X|Y = y) &= -\,\mathbb{E}_{x\leftarrow X} \log \Pr(X = x|Y = y), \\ \mathrm{H}(X|Y) &= \mathbb{E}_{y\leftarrow Y}\, \mathrm{H}(X|Y = y), \\ \mathrm{I}(X : Y|Z) &= \mathrm{H}(X|Z) - \mathrm{H}(X|Y, Z). \end{aligned}$$

The entropy and the mutual information are non negative and satisfy $\mathrm{I}(X : Y|Z) = \mathrm{I}(Y : X|Z)$. For $0 \leq \varepsilon \leq 1$, we denote by $\mathrm{H}(\varepsilon)$ the entropy of the Bernoulli variable that takes value 1 with probability ε, and 0 with probability $1 - \varepsilon$.

3 Reversal with Two Streams

3.1 Complexity in the Read-only/Write-only Model

When $\Sigma = \{0, 1\}$, the naive algorithm, that copies at each pass s bits of the input in memory and then to the output in reverse order, requires memory space $s = n/p$ when performing p passes. When processing each stream for the first time, it is in fact obvious that $Y[i]$ cannot be written until $X[n - i + 1]$ has been read. In particular, for $p = 2$ we have $s \geq n$ for any algorithm. With multiple passes on each stream, the proof gets more technical especially since the point where the streams cross, i. e. where we can write on the second stream what was read on the first one during the current pass depends on the pass, the input and the randomness. However the $s \geq \mathrm{O}(n/p)$ bound still applies for all p, when the error probability of the algorithm is $\mathrm{O}(1/p)$.

Observe that a constant error probability can be reduced to $\mathrm{O}(1/p)$ using $\mathrm{O}(\log p)$ parallel repetitions, leading to an $\mathrm{O}(\log p)$ factor in the size of the memory space.

▶ **Theorem 3.** *Let $0 \leq \varepsilon \leq 1/10$. Every p-pass randomized RO/WO algorithm for* Reverse *on $\{0, 1\}^n$ with error ε requires space $\Omega(n/p)$.*

Expansion is not mentionned in this theorem as it does not make sense on streams that are not read-write: the algorithm either cannot modify the stream if it is read-only, or has no use for the extra data if it is write-only.

Before proving Theorem 3, we begin with two useful facts.

▶ **Fact 4.** *Let X be uniformly distributed over $\{0, 1\}^n$, and let J be some random variable on $\{0, 1, \ldots, n\}$ that may depend on X. Then :*

$$\mathrm{H}(X[1, J]|J) \leq \mathbb{E}(J) \text{ and } \mathrm{H}(X[1, J]|X[J + 1, n]) \geq \mathbb{E}(J) - \mathrm{H}(J).$$

Similarly,

$$\mathrm{H}(X[J + 1, n]|J) \leq n - \mathbb{E}(J) \text{ and } \mathrm{H}(X[J + 1, n]|X[1, J]) \geq n - \mathbb{E}(J) - \mathrm{H}(J).$$

Proof. $\mathrm{H}(X[1, J]|J) \leq \mathbb{E}(J)$ and $\mathrm{H}(X[J + 1, n]|J) \leq n - \mathbb{E}(J)$ are direct. The second part uses the first one as follows:

$$\begin{aligned} \mathrm{H}(X[1, J]|X[J + 1, n]) &= \mathrm{H}(X|J, X[J + 1, n]) - \mathrm{H}(X[J + 1, n]|J, X[J + 1, n]) \\ &= \mathrm{H}(X|J) - \mathrm{H}(X[J + 1, n]|J) \\ &\geq \mathrm{H}(X) - \mathrm{H}(J) - n + \mathbb{E}(J) = \mathbb{E}(J) - \mathrm{H}(J). \end{aligned}$$

◀

▶ **Fact 5** (Data processing inequality). *Let X, Y, Z, R be random variables such that R is independent from X, Y, Z. Then for every function f*

$$\mathrm{H}(X|Y,Z) \leq \mathrm{H}(f(X,R)|Y,Z) \quad \textit{and} \quad \mathrm{I}(X:Y|Z) \geq \mathrm{I}(f(X,R):Y|Z).$$

Note that the previous property is usually stated with no variable R, then f is defined as a *probabilistic function.*

Proof of Theorem 3. In this proof, we use the equivalent model of the algorithm copying X (and not Reverse(X)) on Y, but processing Y only with passes from right to left. This means that the two heads start on opposite ends and meet once each pass.

Consider a p-pass randomized RO/WO algorithm for Reverse on $\{0,1\}^n$ with error ε and space $s \geq \log n$. For simplicity, we assume that passes are synchronized : whenever a pass on one stream ends, the head on the other stream ends its own pass, and then eventually moves back to its original position. This costs us at most a factor 2 in the number of passes.

Let the input stream X be uniformly distributed in $\{0,1\}^n$. For each pass $1 \leq t \leq p$, let $Z^t \in \{0,1,\perp\}^n$ be the reverse of the data written on the output stream Y: if nothing is written at pass t and index i, then $Z^t[i] = \perp$. This model corresponds to writing the same string on the output stream as on the input stream, but going in different directions, an makes the notations simpler as we want $X[i] = Z^t[i]$. Note that because the algorithm cannot overwrite a letter, for each i there is at most one t such that $Z^t[i] \neq \perp$. Last, let L^t be the index where the reading head and the writing head meet during pass t. Since passes are synchronized, L^t is unique (but possibly depends on the input and random choices). For $1 \leq i \leq n$, $1 \leq t \leq p$, let M_i^t be the state of the memory after the algorithm reads $X[i]$ on pass t.

For $1 \leq t \leq p$, since s bounds the size of memory, we have :

$$s \geq \mathrm{I}(X[1,L^t] : M_{L^t}^t | X[L^t+1,n])) \text{ and } s \geq \mathrm{I}(X[L^t+1,n] : M_n^{t+1} | X[1,L^t])).$$

Using the definition of mutual information, we get the following inequalities:

$$s \geq \mathrm{H}(X[1,L^t]|X[L^t+1,n]) - \mathrm{H}(X[1,L^t]|M_{L^t}^t, X[L^t+1,n]),$$

$$s \geq \mathrm{H}(X[L^t+1,n]|X[1,L^t]) - \mathrm{H}(X[L^t+1,n]|M_n^{t-1}, X[1,L^t]).$$

We define the following probabilities : $q_i^t(l) = \Pr(Z^t[i] \neq \perp | L^t = l)$, $q_i^t = \Pr(Z^t[i] \neq \perp)$, $\varepsilon_i^t(l) = \Pr(Z^t[i] \neq \perp, Z^t[i] \neq X[i] | L^t = l)$ and $\varepsilon_i^t = \Pr(Z^t[i] \neq \perp, Z^t[i] \neq X[i])$. By definition, they also satisfy $\varepsilon_i^t = \mathbb{E}_{l \sim L^t}(\varepsilon_i^t(l))$ and $q_i^t = \mathbb{E}_{l \sim L^t}(q_i^t(l))$. Note that by hypothesis[1] and because there is no rewriting, $\sum_{t=1}^p \varepsilon_i^t \leq \Pr(\exists t, Z^t[i] \neq \perp, Z^t[i] \neq X[i]) \leq \varepsilon$. Lemmas 6 and 7 give us these inequalities:

$$2s \geq n - \sum_{i=1}^n \mathrm{H}(X[i]|Z^t[i], L^t) - O(\log n),$$

$$\mathrm{H}(X[i]|Z^t[i], L^t) \leq 1 - q_i^t \left(1 - \mathrm{H}(\varepsilon_i^t / q_i^t)\right).$$

Combining them yields :

$$2s \geq \sum_{i=1}^n q_i^t \left(1 - \mathrm{H}\left((\varepsilon_i^t / q_i^t)\right)\right) - O(\log n).$$

[1] Here we only need the hypothesis that each bit of Y is wrong with probability at most ε, and not the stronger hypothesis that $Y \neq \text{Reverse}(X)$ with probability at most ε.

Let $\alpha_i = \sum_{t=1}^{p} q_i^t$. Then $\alpha_i = \Pr[Y[i] \neq \perp]$ satisfies $\alpha_i \geq 1 - \varepsilon$ by hypothesis. Now summing over all passes leads to $2ps \geq \sum_{i=1}^{n} \alpha_i \sum_{t=1}^{p} (q_i^t/\alpha_i)(1 - \mathrm{H}(\varepsilon_i^t/q_i^t)) - \mathrm{O}(p \log n)$.

The concavity of H gives us $\sum_{t=1}^{p} (q_i^t/\alpha_i)\mathrm{H}(\varepsilon_i^t/q_i^t) \leq \mathrm{H}(\varepsilon/(1-\varepsilon))$. This means, replacing α_i and ε_i^t by their upper bounds, that $2ps \geq n(1-\varepsilon)(1 - \mathrm{H}(\varepsilon/(1-\varepsilon))) - \mathrm{O}(p \log n)$. Since Theorem 3 has $\varepsilon \leq 0.1$ as an hypothesis, our algorithm verifies $ps \geq \Omega(n)$. ◀

▶ **Lemma 6.** *Assuming the hypotheses of Theorem 3, at any given pass t,*

$$2s \geq n - \sum_{i=1}^{n} \mathrm{H}(X[i]|Z^t[i], L) - O(\log n).$$

Proof. In this proof, we write $Z[i]$ for $Z^t[i]$ since there is generally no ambiguity. We similarly omit the t on other notations.

The data processing inequality (Fact 5) gives us the following inequality : $\mathrm{H}(X[1,L]|M_L, X[L+1,n], L) \leq \mathrm{H}(X[1,L]|Z[1,L], L)$. We can rewrite this as

$$\mathrm{H}(X[1,L]|M_L, X[L+1,n], L) \leq \mathbb{E}_{l \sim L}(\mathrm{H}(X[1,l]|Z[1,l], L = l)).$$

Using the chain rule and removing conditioning, we get

$$\mathrm{H}(X[1,L]|M_L, X[L+1,n], L) \leq \mathbb{E}_{l \sim L}(\sum_{i=1}^{l} \mathrm{H}(X[i]|Z[i], L = l)).$$

Similarly,

$$\mathrm{H}(X[L+1,n]|M_n^{t-1}, X[1,L], L) \leq \mathbb{E}_{l \sim L}(\sum_{i=1}^{l} \mathrm{H}(X[i]|Z[i], L = l)).$$

Using that both M_L (where $M_L = M_{L^t}^t$) and M_n^{t-1} are of size at most s bits, we get

$$2s \geq \mathrm{I}(X[1,L] : M_{L^t}^t | X[L+1,n]) + \mathrm{I}(X[L+1,n] : M_n^{t-1} | X[1,L]).$$

Then we conclude by combining the above inequalities and using Fact 4 as follows:

$$\begin{aligned}
2s &\geq \mathbb{E}(L) - \mathrm{H}(L) + n - \mathbb{E}(L) - \mathrm{H}(L) \\
&\quad -\mathrm{H}(X[1,L]|M_L^t, X[L+1,n]) - \mathrm{H}(X[L+1,n]|M_n^{t-1}, X[1,L]) \\
&\geq n - \mathbb{E}_{l \sim L}\left(\sum_{i=1}^{l} \mathrm{H}(X[i]|Z[i], L = l) + \sum_{i=l+1}^{n} \mathrm{H}(X[i]|Z[i], L = l)\right) - \mathrm{O}(\log n) \\
&= n - \sum_{i=1}^{n} \mathrm{H}(X[i]|Z[i], L) - O(\log n).
\end{aligned}$$

◀

▶ **Lemma 7.** *Assuming the hypotheses of Theorem 3, for any pass t,* $\mathrm{H}(X[i]|Z^t[i], L_t) \leq 1 - q_i^t(1 - \mathrm{H}(\varepsilon_i^t/q_i^t))$

Proof. As above, we omit the t in the proof, as there is no ambiguity.

The statement has some similarities with Fano's inequality. Due to the specificities of our context, we have to revisit its proof as follows. First we write

$$\begin{aligned}
\mathrm{H}(X[i]|Z[i],L) &\leq \mathbb{E}_{l\sim L}(\mathrm{H}(X[i]|Z[i],L=l)) \\
&\leq \mathbb{E}_{l\sim L}(q_i(l)\mathrm{H}(X[i]|Z[i],L=l,Z[i]\neq\perp) \\
&\quad +(1-q_i(l))\mathrm{H}(X[i]|L=l,Z[i]=\perp) \\
&\leq \mathbb{E}_{l\sim L}\left(q_i(l)\mathrm{H}\left(\frac{\varepsilon_i(l)}{q_i(l)}\right)+1-q_i(l)\right) \\
&= 1-q_i+\mathbb{E}_{l\sim L}\left(q_i(l)\mathrm{H}\left(\frac{\varepsilon_i(l)}{q_i(l)}\right)\right). \qquad (1)
\end{aligned}$$

By replacing the entropy with its definition, we can see that for any $1\geq q\geq\varepsilon>0$, we have $q\mathrm{H}\left(\frac{\varepsilon}{q}\right)=\mathrm{H}(q-\varepsilon,\varepsilon,1-q)-\mathrm{H}(q)$, where $\mathrm{H}(x,y,z)$ is the entropy of a random variable R in $\{0,1,2\}$ with $\Pr(R=0)=x$, $\Pr(R=1)=y$ and $\Pr(R=2)=z$. Let R_i be such that $R_i=0$ if $X[i]=Z[i]$, $R_i=2$ if $Z[i]=\perp$ and $R_i=1$ otherwise. Note that $(Z[i]=\perp)$ is a function of R_i. Therefore:

$$\begin{aligned}
\mathbb{E}_{l\sim L}\left(q_i(l)\mathrm{H}\left(\frac{\varepsilon_i(l)}{q_i(l)}\right)\right) &= \mathbb{E}_{l\sim L}(\mathrm{H}(R_i|L=l)-\mathrm{H}((Z[i]=\perp)|L=l)) \\
&= \mathrm{H}(R_i|L)-\mathrm{H}((Z[i]=\perp)|L) \\
&= \mathrm{H}(R_i)-\mathrm{H}((Z[i]=\perp))+\mathrm{I}((Z[i]=\perp)|L)-\mathrm{I}(R_i|L).
\end{aligned}$$

By the data processing inequality, $\mathrm{I}((Z[i]=\perp):L)\leq\mathrm{I}(R_i:L)$, so

$$\mathbb{E}_{l\sim L}(q_i(l)\mathrm{H}(\varepsilon_i(l)/q_i(l))))\leq q_i\mathrm{H}(\varepsilon_i/q_i)).$$

Combining this with inequality 1 gives us the lemma. ◀

3.2 Complexity of Read-only/Read-write and Read-write/Write-only

In this section, we prove tight bounds in the Read-only/Read-write and Read-write/Write-only model. These models are more complex than the previous one since an algorithm may now modify $Y[i]$ or $X[i]$ several times, and use that as additional memory.

Algorithm. As a subroutine we use Algorithm 1, which performs $\mathrm{O}(\sqrt{n})$ passes and uses space $\mathrm{O}(\log|\Sigma|)$. It works by copying blocks of size $\mathrm{O}(\sqrt{n})$ from the input stream to the output stream without reversing them (otherwise there would not be enough space in the memory), but putting them in the correct order pairwise. In addition, during each pass on the output, it moves one element from each block already copied to the correct place. Since blocks have as many elements as there are passes left after they are copied, the output stream is in the correct order at the end of the execution .

▶ **Theorem 8.** *There is a deterministic algorithm such that, given n and $p\leq\sqrt{n}$, it is a p-pass RO/RW streaming algorithm for* Reverse *on Σ^n with space $\mathrm{O}(\log n+(n\log|\Sigma|)/p^2)$ and expansion* 1.

Proof. We will prove that Algorithm 1 satisfies the theorem when $p=2\sqrt{n}$. For the general case, the algorithm treats groups of $m=4n/p^2$ letters as though they were just one letter of the new alphabet Σ^m, and then runs Algorithm 1. This uses spaces $\mathrm{O}(\log(n/m)+(n/m)(\log|\Sigma^m|)/p^2)=\mathrm{O}(\log n+(n\log|\Sigma|)/p^2)$.

Algorithm 1 RO/RW streaming algorithm for Reverse.

```
1  p ← √(2n), t ← 1, i_1 ← p;
2  While {t ≤ p}
3    If t > 1 then (R, l) ← (Y[n − i_t], n − i_t)
4    If i_t < n then
5      Y[n − i_t − (p − t), n − i_t] ← X[i_t − (p − t), i_t] // Order is unchanged
6      i_{t+1} ← i_t + p − t
7    For m = t − 1 to 1
8      Put R in the right place Y[l'] // l' computed from l, m, p, n
9      (R, l) ← (Y[l'], l')
10   t ← t + 1
```

We now prove the theorem for $p = 2\sqrt{n}$. First, observe that every element of the input is copied on the output, as $\sum_{t=1}^{p}(p-t) = p(p+1)/2 = n + \sqrt{n} \geq n$.

Let $1 \leq t < p$. The subword $X[i_t - (p-t) - 1, i_t]$ is initially copied at line 5 on $Y[n - i_t - (p-t) - 1, n - i_t]$. Therefore only $Y[n - i_t]$ is correctly placed. Let B_t be $\{n - i_t - (p-t), n - i_t - 1\} = \{n - i_{t+1} + 1, n - i_t - 1\}$. Then B_t denotes the indices (on Y) of elements copied during the t-th iteration of the While loop that are incorrectly placed. For each $l \in B_t$, the correct place for $Y[l]$ is in $\{n - i_t + 1, n - i_t + p - t\} = \{n - i_t + 1, n - (i_{t-1} + 1)\} = B_{t-1}$, where by convention B_0 is defined with $i_0 = 0$. Therefore, in the $(t+1)$-th iteration of the while loop, the first $Y[l]$ we place correctly goes from $l = n - i_{t+1} + 1 = n - i_t - (p-t) \in B_t$ to some $l' \in B_{t-1}$. Then recursively the previous value of $Y[l']$ goes in B_{t-2}, and so on until we reach B_0 where nothing was written initially.

Thus, the For loop places correctly one element of each of B_{t-1}, $B_{t-2}, \ldots, B_1$. Observe that B_m has at most $p - m$ elements incorrectly placed. Moreover, there are $(p - m)$ remaining iterations of the While loop after B_m is written. Therfore all elements are places correctly when Algorithm 1 ends.

Now we prove that Algorithm 1 has the claimed complexity. It only starts a new pass when t increases, at each execution of the While loop at line 2. Indeed, each execution of line 8 can be performed within the current pass since $Y[l]$ moves forward to a new index in B_{m-1} as explained before. Therefore Algorithm 1 runs in p passes. Since it keeps at most two elements in memory at any time, and only needs to keep track of the current position, n, p, t and m, it uses $O(\log n + \log |\Sigma|)$ memory. ◀

▶ **Theorem 9.** *There is a deterministic algorithm such that, given n and $p \leq \sqrt{n}$, it is a p-pass RW/WO streaming algorithm for* Reverse *on Σ^n with space* $\mathrm{O}(\log n + (n \log |\Sigma|)/p^2)$ *and expansion* 1.

We omit the proof of this theorem, as the algorithm is extremely similar to the one used in the Read-only/Read-write model. The main difference is that the For loop that moves one element of each block to its correct place is applied before a block is copied on Y and not after like in Algorithm 1. Similarly, the new algorithm starts with blocks of size s and ends with size $\Theta(\sqrt{n/s})$ instead of having blocks of decreasing size.

Lower Bound. We employ techniques similar to the ones we used in the proof of Theorem 3. However, here we will not consider individual cells on the stream but instead blocks of size $k = \sqrt{ns}$, where s is the memory space. This allows us to easily bound the amount of information each block receives.

▶ **Theorem 10.** *Let $0 < \varepsilon \leq 1/3$. Every p-pass λ-expansion RO/RW streaming algorithm for* Reverse *on $\{0,1\}^n$ with error ε requires space $\Omega(n/p^2)$.*

Proof. Consider a p-pass λ-expansion randomized RO/RW algorithm for Reverse on $\{0,1\}^n$ with error ε and space s, with $s \geq \log n$. Like with Theorem 3, we consider the model where we want $Y = X$, but Y is processed from right to left.

As in the proof of Theorem 3, we assume passes are synchronized at the cost of a factor at most 2 in p. We also keep similar notations : X is the input stream uniformly distributed in $\{0,1\}^n$, and for each $1 \leq t \leq p$, $Y^t \in \{0,1,\perp\}^n$ is the data currently on output stream Y at pass t. Unlike with Z^t in the previous section, this includes the data previously written, as in this model we can read it and modify it. Let $1 \leq k \leq n$ be some parameter. We now think on X, Y^t as sequences of k blocks of size n/k, and consider each block as a symbol. If $\lambda > 1$, then everything written on the output stream (the only one that can grow) after the n-th bit is considered to be part of the Y_k^t. For instance X_i denotes the i-th block of X, which is of size n/k bits. We write X_{-i} for X without its i-th block, and $X_{>i}$ (resp. $X_{<i}$) for the last $(k-i)$ blocks of X (resp. the first $(i-1)$ blocks). For each $1 \leq t \leq p$, let $L_t \in \{0,\dots,k-1\}$ be the block where the input head and the output head meet during the t-th pass. Since passes are synchronised, L_t is unique and is the only block where both heads can be simultaneously during the t-th pass. Let M_i^t be the memory state as the output head enters the i-th block during t-th pass.

Consider a pass t and a block i. We would like to have an upper bound on the amount of mutual information between Y_i^t and X_i that is gained during pass t (with regards to information known at pass $t-1$). Let $\Delta_{i,j}^t = \mathrm{I}(X_i : Y_i^t | L_t = j, X_{-i}) - \mathrm{I}(X_i : Y_i^{t-1} | L_t = j, X_{-i})$ for some i and j. Of course, if $i = j$, without looking inside the block structure we only have the trivial bound $\Delta_{i,i}^t \leq H(X_i) = n/k$. It is however easier to bound other blocks. Assume without loss of generality that $i < j$. We use the data processing inequality $I(f(A) : B|C) \leq I(A : B|C)$ with Y_i^t as a function of M_i^t and Y_i^{t-1}. This gives us

$$\Delta_{i,j}^t \leq \mathrm{I}(X_i : X_{>i}, M_i^t, Y_i^{t-1} | L_t = j, X_{-i}) - \mathrm{I}(X_i : Y_i^{t-1} | L_t = j, X_{-i}).$$

We can remove $X_{>i}$ which is contained in the conditioning. Applying the chain rule, we cancel out the second term and are left with

$$\Delta_{i,j}^t \leq \mathrm{I}(X_i : M_i^t | L_t = j, X_{-i}, Y_i^{t-1}) \leq \mathrm{H}(M_i^t) \leq s.$$

The same holds if $j < i$ instead. If $j = i$, then $\Delta_{i,j}^t \leq n/k$ because $\mathrm{H}(X_i) = n/k$.

We fix j, sum over i and get $\sum_{i=0}^{k-1} \Delta_{i,j}^t \leq n/k + ks$. The expectation over $j \sim L_t$ is

$$\sum_{i=0}^{k-1} \mathrm{I}(X_i : Y_i^t | L_t, X_{-i}) - \mathrm{I}(X_i : Y_i^{t-1} | L_t, X_{-i}) \leq n/k + ks.$$

From Fact 11, we get the following inequalities:

$$\mathrm{I}(X_i : Y_i^t | L_t, X_{-i}) \geq \mathrm{I}(X_i : Y_i^t | X_{-i}) - \mathrm{H}(L_t),$$

$$\mathrm{I}(X_i : Y_i^{t-1} | L_t, X_{-i}) \leq \mathrm{I}(X_i : Y_i^{t-1} | X_{-i}) + \mathrm{H}(L_t).$$

Therefore,

$$\sum_{i=0}^{k-1} \mathrm{I}(X_i : Y_i^t | X_{-i}) - \mathrm{I}(X_i : Y_i^{t-1} | X_{-i}) \leq n/k + ks + k \log k.$$

Summing over t yields

$$\sum_{i=0}^{k-1} \mathrm{I}(X_i : Y_i^p | X_{-i}) \leq p(n/k + ks + k \log k).$$

By hypothesis $s \geq \log n$, $\varepsilon \leq 1/3$ and $\mathrm{I}(X_i : Y_i^p | X_{-i}) \geq (1 - \mathrm{H}(\varepsilon))n/k$. If $k = \sqrt{n/s}$, then $s = \Omega(n/p^2)$. ◀

▶ **Fact 11.** *Let A, B, C, D be random variables. Then*

$$\mathrm{I}(A : B|D) - \mathrm{H}(C|D) \leq \mathrm{I}(A : B|C, D) \leq \mathrm{I}(A : B|D) + \mathrm{H}(C|D).$$

▶ **Theorem 12.** *Let $0 < \varepsilon \leq 1/3$. Every p-pass λ-expansion RW/WO streaming algorithm for* Reverse *on $\{0,1\}^n$ with error ε requires space $\Omega(n/p^2)$.*

Proof of Theorem 12. Consider a p-pass λ-expansion randomized RW/WO algorithm for Reverse on $\{0,1\}^n$ with error ε and space s, with $s \geq \log n$. As before, we consider the model where the algorithm writes X on Y, using passes from right to left. This prrof is similar to the proof of Theorem 10.

We proceed as before: we assume passes are synchronized at the cost of a factor at most 2 in p. We also keep similar notations : X is the input stream uniformly distributed in $\{0,1\}^n$, and for each $1 \leq t \leq p$, $X^t \in \{0,1\}^n$ is the content of the input stream and $Y^t \in \{0,1,\perp\}^n$ is the content of the output stream Y at pass t. Let $1 \leq k \leq n$ be some parameter. As before, whe think of X, X^t, Y^t as sequences of k blocks of size n/k, and X_i denotes the i-th block of X. If $\lambda > 1$, then everithing written on the inpu stream after the n-th bit is considered part of X_k^t. We write X_{-i}, $X_{>i}$ and $X_{<i}$ as before. We also define $X_i^{\leq t}$ as the $(t+1)$-uple $(X_i, X_i^1, \ldots, X_i^t)$, i. e. the history of the i-th block until pass t.

As in prevoius proofs, for each $1 \leq t \leq p$, let $L_t \in \{0, \ldots, k-1\}$ be the block where the input head and the output head meet during the t-th pass. L_t is unique and is the only block where both heads can be simultaneously during the t-th pass. Let M_i^t be the memory state as the output head enters the i-th block during t-th pass.

Consider a pass t and a block i. As with Theorem 10, we would like to have an upper bound on the amount of mutual information between Y_i^t and X_i that is gained during pass t, assuming $L_t \neq i$. Let $\Delta_{i,j}^t = \mathrm{I}(X_i : Y_i^t | L_t = j, X_{-i}^{\leq t-1}) - \mathrm{I}(X_i : Y_i^{t-1} | L_t = j, X_{-i}^{\leq t-1})$ for some $j > i$. By the data processing inequality, we have $I(f(A) : B|C) \leq I(A : B|C)$. Therefore,

$$\Delta_{i,j}^t \leq \mathrm{I}(X_i : X_{>i}^{t-1}, M_i^t, Y_i^{t-1} | L_t = j, X_{-i}^{\leq t-1}) - \mathrm{I}(X_i : Y_i^{t-1} | L_t = j, X_{-i}^{\leq t-1}).$$

We can remove $X_{>i}^{t-1}$ which is contained in the conditioning. Applying the chain rule, we cancel out the second term and are left with

$$\Delta_{i,j}^t \leq \mathrm{I}(X_i : M_i^t | L_t = j, X_{-i}^{\leq t-1}, Y_i^{t-1}) \leq \mathrm{H}(M_i^t) \leq s.$$

The same holds if $j < i$ instead. If $j = i$, then $\Delta_{i,j}^t \leq n/k$ because $\mathrm{H}(X_i) = n/k$.

We fix j, sum over i and get $\sum_{i=0}^{k-1} \Delta_{i,j}^t \leq n/k + ks$. As before, by taking the expectation over $j \sim L_t$ and then using Fact 11, we can remove the condition $L_t = j$. This gives us

$$\sum_{i=0}^{k-1} \mathrm{I}(X_i : Y_i^t | X_{-i}^{\leq t-1}) - \mathrm{I}(X_i : Y_i^{t-1} | X_{-i}^{\leq t-1}) \leq n/k + ks + k \log k.$$

Algorithm 2 RW/RW streaming algorithm for Reverse

```
1  W_0 ← X ; W_1 ← Y ; α ← 0 ; // Rename the streams
2  k ← n ; // Size of current blocks
3  While k > 1
4    k ← k/2
5    For i = 1 to n/2k − 1
6      W_{1−α}[(2i + 1)k + 1, (2i + 2)k] ← W_α[2ik + 1, (2i + 1)k] // One pass
7    For i = 0 to n/2k − 1
8      W_{1−α}[2ik + 1, (2i + 1)k] ← W_α[(2i + 1)k + 1, (2i + 2)k] // Another pass
9    α ← 1 − α ; Erase W_{1−α} ; // Exchange the roles of the streams
10 Y ← W_α // Copy the final result on output tape
```

We cannot sum over t yet because the conditionning $X_{-i}^{\leq t-1}$ depends on t. However, because X_{-i}^{t} is a function of X_{-i}^{t-1}, the memory state at the beginning of the pass and the memory as the head on the input tape leaves the i-th block, applying Fact 11 again yields

$$\mathrm{I}(X_i : Y_i^t | X_{-i}^{\leq t}) - \mathrm{I}(X_i : Y_i^t | X_{-i}^{\leq t-1}) \leq \mathrm{H}(X_{-i}^t | X_{-i}^{\leq t-1}) \leq 2s.$$

This is a consequence of the output stream being write-only, which we had not used until now.

Therefore $\sum_{i=0}^{k-1} \mathrm{I}(X_i : Y_i^t | X_{-i}^{\leq t-1}) - \mathrm{I}(X_i : Y_i^{t-1} | X_{-i}^{\leq t-1}) \leq n/k + 3ks + k \log k$. Summing over t yields

$$\sum_{i=0}^{k-1} \mathrm{I}(X_i : Y_i^p | X_{-i}) \leq p(n/k + 3ks + k \log k).$$

By hypothesis $s \geq \log n$, $\varepsilon \leq 1/3$ and $\mathrm{I}(X_i : Y_i^p | X_{-i}) \geq (1 - \mathrm{H}(\varepsilon))n/k$. If $k = \sqrt{n/s}$, then $s = \Omega(n/p^2)$. ◀

Note that the proof still works if we relax the write-only model by allowing the algorithm to rewrite over data that was previously written on the output stream.

3.3 Complexity of Read-write/Read-write

Algorithm 2 proceeds by dichotomy. For simplicity, we assume that n is a power of 2, but the algorithm can easily be adapted while keeping $\lambda = 1$. At each step, it splits the input in two, copies one half to its correct place on the stream, then makes another pass to copy the other half, effectively exchanging them.

▶ **Theorem 13.** *Algorithm 2 is a deterministic $O(\log n)$-passes RW/RW streaming algorithm for* Reverse *on Σ^n with space $\mathrm{O}(\log n)$ and expansion 1.*

Proof. Since the algorithm can read and write on both tapes, they perform very similar roles. We rename the input stream W_0 and the output stream W_1. By a simple recursion, we see that whenever a block $W_{1-\alpha}[tk, (t+1)k]$ is moved, it is moved in the place that $\mathrm{Reverse}(W_{1-\alpha}[tk, (t+1)k])$ will occupy. Therefore, the algorithm is correct.

Now we prove the bounds on s and p. Algorithm 2 never needs to remember a value, only the current index and current pass, so $s = \mathrm{O}(\log n)$. Since the length of blocks copied is divided by 2 at each execution of line 4, it ends after a logarithmic number of executions of the While loop. Each iteration of the While loop requires two passes, one for each of the two For loops (lines 5 and 7). Therefore the total number of passes is in $\mathrm{O}(\log n)$. ◀

Algorithm 3 RW/RW streaming algorithm implementing Merge Sort

```
1  W_0 ← X ; W_1 ← Y ; α ← 0 ;
2  t ← 1 ; k ← 1 // Size of the sorted blocks
3  Expand the input : each element has a label of size log n.
4  While k < n
5    For i = 1 to n/2k {Copy B^t_{2i} on W_{1−α}}
6    For i = 1 to n/2k {For each W_α[j] ∈ B^t_{2i−1} ∪ B^t_{2i}
7                       {W_α[j] ← index of W_α[j] in B^{t+1}_i}}
8    For i = 1 to n/2k {Copy B^t_{2i} at the end of W_α after B^t_{2i−1}}
9    For i = 1 to n/2k {For each W_α[j] ∈ B^t_{2i−1}
10                      {Write W_α[j] at its position on W_{1−α}}}
11   For i = 1 to n/2k {For each W_α[j] ∈ B^t_{2i}
12                      {Write W_α[j] at its position on W_{1−α}}}
13   α ← 1 − α ; Erase W_{1−α} ; t ← t + 1 ; k ← 2k ;
14 Y ← W_α
```

4 Sorting with Two Streams

Sort is generally more complex than Reverse. Even in the RW/RW model, we are not able to present a deterministic algorithm as efficient as Algorithm 2 for Reverse. With three streams, the problem becomes easy since we can Merge Sort two streams, and write the result of each step on the third one.

4.1 Merge Sort

We begin with an algorithm inspired from [5]. The algorithm works as a Merge Sort. We call B_i^t the i-th sorted block at the t-th iteration of the While loop, consisting of the sorted values of $X[2^t i + 1, 2^t(i+1)]$. Since there is no third stream to write on when two blocks B_{2i-1}^t and B_{2i}^t are merged into B_i^{t+1}, we label each element with its position in the new block. Then both halves are copied on the same stream again so that they can be merged with the help of the labels. This improves the expansion of [5] from n to $\log n$. However it is somewhat unsatisfying because when Σ is of constant size, our algorithm still has $\Omega(\log n)$ expansion.

▶ **Theorem 14.** *Algorithm 3 is a deterministic* $\mathrm{O}(\log n)$*-pass RW/RW streaming algorithm for* Sort *on* Σ^n *with space* $\mathrm{O}(\log n)$ *and expansion* $\mathrm{O}((\log n)/\log|\Sigma|)$.

Proof. Since it is an implementation of the Merge Sort algorithm, Algorithm 3 is correct. Each iteration of the While loop corresponds to five passes on each tape, and therefore the total number of passes is in $\mathrm{O}(\log n)$. Since the algorithm only needs to remember the position of the heads, current elements and the counters k, t, it only uses memory $\mathrm{O}(\log n)$. Finally, since the label for each element is at most n, we only use space $\log n$ on the stream to write it, and therefore the expansion is at most $(\log|\Sigma| + \log n)/\log|\Sigma| = \mathrm{O}((\log n)/\log|\Sigma|)$. ◀

4.2 Quick Sort

With a Quick Sort algorithm instead of a Merge Sort, we only need to store the current pivot (of size at most $\log n$), without labeling elements. However, Quick Sort comes with its own issues: the expected number of executions of the While loop is $\mathrm{O}(\log n)$, but unless we can select a good pivot it is $\Omega(n)$ in the worst case. For this reason, we use a randomized Las vegas algorithm.

Algorithm 4 RW/RW streaming algorithm implementing Quick Sort

```
1  W_0 ← X ; W_1 ← Y ; α ← 0 ;
2  t ← 1 ; K ← 1 // Number of unsorted blocks
3  While K > 0
4    Abort if the total number of passes is Ω((log n)/ε)
5    Expand W_0 adding O(K) space for ♯ and pivots
6    For i = 1 to K
7      Find P_i^t at random
8      W_{1−α}[i] ← P_i^t
9      Replace P_i^t with a ⊥ on W_α
10   For i = 1 to K
11     Copy W_{1−α}[i] = P_i^t at the start of B_i^t on W_α
12   For i = 1 to K
13     Write all elements in B_{2i−1}^{t+1} on W_{1−α}
14     Write ♯P_i♯ on W_{1−α}.
15     Leave space for the rest of B_{2i}^{t+1}
16   For i = 1 to K
17     Write all elements in B_{2i}^{t+1} in the space left on W_{1−α}
18   α ← 1 − α ; Erase W_{1−α} ; t ← t + 1
19   K ← new number of unsorted blocks // using an additional pass
20 Y ← W_α
```

A block in Algorithm 4 is a set of elements that are still pairwise unsorted, i. e. elements that have the same relative positions to all pivots so far. The block B_i^t is the i-th lower one during the t-th iteration of the While loop, and P_i^t is its pivot. The block B_{2i-1}^{t+1} consists of all elements in B_i^t lower than P_i^t and all elements equal P_i^t with a lower index. The block B_{2i}^{t+1} is the complementary. Algorithm 4 marks the borders of blocks with the symbol ♯.

Algorithm 4 selects each pivot P_i^t at random among the elements of B_i^t. While it may not do so uniformly with only one pass because $|B_i^t|$ is unknown, it has an upper bound $k \geq |B_i^t|$. Algorithm 4 selects $l \in \{1, \ldots, k\}$ uniformly at random, then picks l_i the remainder modulo $2^{\lceil \log |B_i^t| \rceil}$ of l. This can be computed in one pass with $O(\log n)$ space by updating l_i as the lower bound on $|B_i^t|$ grows. It uses $P_i^t = B_i^t[l_i]$. P_i^t is selected uniformly at random from a subset of size at least half of B_i^t, which guarantees that with high probability $min(|B_{2i-1}^{t+1}|, |B_{2i}^t|) = \Omega(|B_i^t|)$.

▶ **Theorem 15.** *For all $0 < \varepsilon \leq 1/2$, Algorithm 4 is a randomized $O((\log n)/\varepsilon)$-pass RW/RW Las Vegas streaming algorithm for Σ^n with failure ε, space $O(\log n)$ and expansion $O(1)$.*

Proof. Algorithm 4 is an implementation of the Quick Sort algorithm, and is therefore correct. Its correctness does not depend on the quality of the pivots. The bound on the memory and the expansion are direct. While it uses additional symbols ⊥ and ♯, they can be easily replaced by an encoding with symbols of Σ appearing in the input with only $O(1)$ expansion. Because for each i and t, with high probability $min(|B_{2i-1}^{t+1}|, |B_{2i}^t|) = \Omega(|B_i^t|)$, with high probability Algorithm 4 terminates in $O(\log n)$ passes. ◀

5 Open Problems

The first open problem is whether our lower bounds still hold if multiple rewrites are allowed in the read-only/write-only model.

Another one is whether there exists a deterministic $O(\log n)$-pass RW/RW streaming algorithm for Sort with space $O(\log n)$ and expansion $O(1)$. We can derandomize Algorithm 4

using any deterministic algorithm that finds a good approximation of the median. The best algorithm we obtained that way has $O(\alpha^{-1} \log n)$ passes, $O(n^\alpha)$ memory and $O(1)$ expansion, for any $\alpha > 0$. We conjecture that there is no such algorithm and that having constant expansion algorithm for Sort requires a tradeoff in number of passes, memory space, number of streams or determinism.

Last, combining the algorithm from Theorem 8 with the results in [14], we obtain a $O(\sqrt{n}/\log n)$-pass RO/RW streaming algorithm with space $O((\log n)^2)$ for recognizing well-parenthesized expressions with two parentheses types. We do not know if this is optimal.

References

1 Alok Aggarwal and Jeffrey Vitter. The input/output complexity of sorting and related problems. *Communications of the ACM*, 31(9):1116–1127, 1988.

2 Paul Beame and Trinh Huynh. The value of multiple read/write streams for approximating frequency moments. *ACM Transactions on Computation Theory*, 3(2):6, 2012.

3 Paul Beame, T.S. Jayram, and Atri Rudra. Lower bounds for randomized read/write stream algorithms. In *Proceedings of the 39th annual ACM symposium on Theory of computing*, pages 689–698, 2007.

4 Amit Chakrabarti, Graham Cormode, Ranganath Kondapally, and Andrew McGregor. Information cost tradeoffs for augmented index and streaming language recognition. *SIAM Journal on Computing*, 42(1):61–83, 2013.

5 Jianer Chen and Chee-Keng Yap. Reversal complexity. *SIAM Journal on Computing*, 20(4):622–638, 1991.

6 Nathanaël François and Frédéric Magniez. Streaming complexity of checking priority queues. In *Proceeding of the 30th International Symposium on Theoretical Aspects of Computer Science*, page 454. Citeseer, 2013.

7 Travis Gagie. On the value of multiple read/write streams for data compression. In *Information Theory, Combinatorics, and Search Theory*, pages 284–297. Springer, 2013.

8 Martin Grohe, André Hernich, and Nicole Schweikardt. Lower bounds for processing data with few random accesses to external memory. *Journal of the ACM*, 56(3):12, 2009.

9 Martin Grohe, Christoph Koch, and Nicole Schweikardt. Tight lower bounds for query processing on streaming and external memory data. *Theoretical Computer Science*, 380(1):199–217, 2007.

10 André Hernich and Nicole Schweikardt. Reversal complexity revisited. *Theoretical Computer Science*, 401(1):191–205, 2008.

11 Rahul Jain and Ashwin Nayak. The space complexity of recognizing well-parenthesized expressions. Technical Report 71, Electronic Colloquium on Computational Complexity, 2010.

12 Tiko Kameda and Roland Vollmar. Note on tape reversal complexity of languages. *Information and Control*, 17(2):203–215, 1970.

13 Christian Konrad and Frédéric Magniez. Validating xml documents in the streaming model with external memory. *ACM Transactions on Database Systems*, 38(4):27, 2013.

14 Frédéric Magniez, Claire Mathieu, and Ashwin Nayak. Recognizing well-parenthesized expressions in the streaming model. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 261–270, 2010.

15 James I. Munro and Mike S. Paterson. Selection and sorting with limited storage. *Theoretical computer science*, 12(3):315–323, 1980.

16 S. Muthukrishnan. *Data streams: Algorithms and applications*. Now Publishers Inc, 2005.

17 Jan Matthias Ruhl. *Efficient algorithms for new computational models*. PhD thesis, Citeseer, 2003.

Improved Lower Bounds for Testing Triangle-freeness in Boolean Functions via Fast Matrix Multiplication*

Hu Fu[1] and Robert Kleinberg[2]

1 **Microsoft Research, New England Lab**
`hufu@microsoft.com`
2 **Cornell University, Department of Computer Science**
`rdk@cs.cornell.edu`

Abstract

Understanding the query complexity for testing linear-invariant properties has been a central open problem in the study of algebraic property testing. Triangle-freeness in Boolean functions is a simple property whose testing complexity is unknown. Three Boolean functions f_1, f_2 and $f_3 : \mathbb{F}_2^k \to \{0,1\}$ are said to be triangle free if there is no $x, y \in \mathbb{F}_2^k$ such that $f_1(x) = f_2(y) = f_3(x+y) = 1$. This property is known to be strongly testable [16], but the number of queries needed is upper-bounded only by a tower of twos whose height is polynomial in $1/\epsilon$, where ϵ is the distance between the tested function triple and triangle-freeness, i. e., the minimum fraction of function values that need to be modified to make the triple triangle free. A lower bound of $(\frac{1}{\epsilon})^{2.423}$ for any one-sided tester was given by Bhattacharyya and Xie (2010). In this work we improve this bound to $(\frac{1}{\epsilon})^{6.619}$. Interestingly, we prove this by way of a combinatorial construction called *uniquely solvable puzzles* that was at the heart of Coppersmith and Winograd (1990)'s renowned matrix multiplication algorithm.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Property testing, linear invariance, fast matrix multiplication, uniquely solvable puzzles

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.669

1 Introduction

Property testing studies algorithms using a small number of queries to a large input that decides, with high probability, whether the input satisfies a certain property or is far from it. Typically, the input f is a function mapping from a finite domain D to a range R. A property $\mathcal{P}$ is a subset of all such functions $\{f : D \to R\}$. If we measure the distance between two functions by the Hamming metric, $\mathrm{dist}(f,g) := \mathbf{Pr}_x[f(x) \neq g(x)]$, then the distance from f to the property $\mathcal{P}$ is $\mathrm{dist}(f,\mathcal{P}) := \min_{g\in\mathcal{P}} \mathrm{dist}(f,g)$. Fixing a distance ϵ, an algorithm, called a *tester*, makes randomized queries to f, and outputs YES with probability at least $2/3$ for $f \in \mathcal{P}$, and NO with probability at least $2/3$ if $\mathrm{dist}(f,\mathcal{P}) \geq \epsilon$. A tester is said to be *one-sided* if it outputs YES with probability one for $f \in \mathcal{P}$. The central question studied by property testing, as initiated by Rubin and Sudan [21] and Goldreich et al. [15], is to

* Hu Fu did part of the work when he was a Ph. D. student at Cornell University, where he was supported by NSF award AF-0910940. Robert Kleinberg was supported in part by NSF awards CCF-0643934 and AF-0910940, AFOSR grant FA9550-09-1-0100, a Microsoft Research New Faculty Fellowship, and a Google Research Grant.

understand the query complexity, i. e., the minimum number of queries needed by a tester, to test various properties.

For example, a property is called *strongly testable* if its query complexity does not depend on the size of the doamin $|D|$ and is only a function of ϵ. For graph and hypergraph properties, strongly testable properties have been exactly characterized [2]. Among strongly testable properties, it is important to understand which ones admit testers with query complexity polynomial in $1/\epsilon$ and which do not. For example, for undirected graphs and one-sided testers, H-freeness for a fixed subgraph H has polynomial query complexity if and only if H is bipartite [1]. Similar characterizations are known for directed graphs and hypergraphs [3, 4, 20, 6].

Kaufman and Sudan (2008) suggested that symmetries, or invariances under transformations of a property, play an important role in facilitating efficient testers. As an easy example, a graph property, seen as a function on graph edges, is invariant under graph isomorphisms, i. e. permutations of the nodes. Kaufman and Sudan launched the systematic study of *algebraic* property testing, and in particular singled out *linear-invariant* properties as a natural class of properties to consider. Restricted to the context of Boolean functions, a property $\mathcal{P} \subset \{f : \mathbb{F}_2^k \to \{0,1\}\}$ is said to be linear-invariant if for all $f \in \mathcal{P}$ and linear transformation $L : \mathbb{F}_2^k \to \{0,1\}$, the composition $f \circ L$ is still in $\mathcal{P}$. One may further define a property $\mathcal{P}$ to be *linear* if it is closed under linear operations; for a property $\mathcal{P}$ on Boolean functions, this simply means $f, g \in \mathcal{P}$ entails $f + g \in \mathcal{P}$. Kaufman and Sudan [18] showed that all properties that are linear-invariant and linear can be tested with query complexity polynomial in $1/\epsilon$. When the linearity condition is relaxed, however, the picture of what is currently understood is less clear. *Triangle-freeness* is one such property.

A function $f : \mathbb{F}_2^k \to \{0,1\}$ is said to be triangle-free if there are no $x, y \in \mathbb{F}_2^k$ such that $f(x) = f(y) = f(x + y) = 1$. More generally, f is said to be (M, σ)-free for a fixed matrix $M \in \mathbb{F}_2^{r \times s}$ and vector $\sigma \in \{0,1\}^s$, if there exists no $x = (x_1, \ldots, x_s) \in (\mathbb{F}_2^k)^s$ such that $Mx = \mathbf{0}$ and $f(x_i) = \sigma_i$ for all $i \in [s]$. Green [16] showed that $(M, \mathbf{1})$-freeness with rank-one matrix M is strongly testable (which includes triangle-freeness), and started the line of investigations resolving that any $(M, \mathbf{1})$-freeness is strongly testable [19, 23], and that the intersection of (possibly infinite) (M, σ)-freeness, with rank-one M, is testable [8, 10]. However, the upper bounds for the number of queries given in these works, though independent of k, are all towers of twos whose heights are polynomial in $1/\epsilon$. The only exception is a result of Bhattacharyya et al. [9] showing that odd-cycle-freeness can be tested with $\tilde{O}(1/\epsilon^2)$ queries. It was noted by Bhattacharyya et al. that this property is the intersection of *infinite* $(M, \mathbf{1})$-freeness. In fact, it has been conjectured that testing any odd cycle alone takes superpolynomial number of queries. Prior to this work, the only nontrivial bound for the simplest such property, triangle-freeness, was given by Bhattacharyya and Xie [11], who showed that any one-sided tester needs $\Omega(1/\epsilon^{2.423})$ queries. This is in sharp contrast with our complete understanding of the query complexity of testing H-freeness in graphs, the counterpart among graph properties to $(M, \mathbf{1})$-freeness.

Our Results

In this work we improve Bhattacharyya and Xie's lower bound [11] and show that any one-sided tester needs $\Omega(1/\epsilon^{6.619})$ queries to test triangle-freeness in Boolean functions. Bhattacharyya and Xie's lower bound was built on families of vectors having a combinatorial property called *perfect-matching-free* (PMF, Definition 3). Roughly speaking, a PMF family can be expanded to construct Boolean functions such that for every x with $f(x) = 1$, there exist a small number of y's such that $f(y) = f(x + y) = 1$. Such a function has a number

of triangles that is about linear with the number of 1's needed to be flipped to remove all triangles. In other words, the number of triangles is relatively small whereas the distance to triangle-freeness is relatively large, a difficult scenario for a tester. However, Bhattacharyya and Xie were able to find only very small (and hence weak) PMF families by way of numerical calculations. When the dimension of the family exceeds 5 the calculation becomes forbiddingly expensive.

In this work, we are able to construct large PMF families by using a combinatorial structure called *uniquely solvable puzzles* (USP, Definition 5). USPs were defined by Cohn et al. [12] in their group theoretic approach to fast matrix multiplication. Under their perspective, the most important step in Coppersmith and Winograd (1990)'s famous $O(n^{2.376})$-time algorithm for multiplication of $n \times n$ matrices was a construction of large USPs. Coppersmith and Winograd"s algorithm was for a long time the best known algorithm for this fundamental problem, and was improved only recently [24, 25]. As we recall in Appendix A, Coppersmith and Winograd's construction crucially relies on large sets of densely populated integers with no three terms in arithmetic progressions [22, 7, 14]. Seen through the connection we identify here, it may not be a coincidence that the superpolynomial lower bounds for testing nonbipartite H-freeness in graphs also crucially used such sets with no arithmetic progressions [1]. However, we were unable to give superpolynomial lower bounds for testing triangle-freeness in Boolean functions.

This leads to some fascinating open problems. For example, Cohn et al. [12] showed that, if large families of a strengthened version of USPs, called strongly uniquely solvable puzzles (SUSP), exist, then the exponent of matrix multiplication is 2, as has long been conjectured. Would a large SUSP imply superpolynomial query complexity for testing any $(M, \mathbf{1})$-freeness in Boolean functions? On the other hand, would such a lower bound imply the success of Cohn et al.'s campaign on matrix multiplication? We leave these questions for future investigation. These questions are all the more intriguing, given connections between SUSPs and sunflower conjecutres (recently established by Alon et al. [5]) and connections between sunflower conjectures and large PMF constructions (discovered by Haviv and Xie [17] following our work)—a possible link between SUSPs and PMFs is still missing.

2 Preliminaries

For an integer n, we let $[n]$ denote the set $\{1, 2, \ldots, n\}$. We use $\mathrm{Sym}(S)$ to denote the symmetric group on a set S. We will often identify a Boolean function $f : \mathbb{F}_2^k \to \{0, 1\}$ with the family of subsets in $[k]$ whose indicator function is f.

We will focus on testing triangle-freeness for Boolean function triples.[1] A function triple $f_1, f_2, f_3 : \mathbb{F}_2^k \to \{0, 1\}$ is said to be triangle-free if there is no $x, y \in \mathbb{F}_2^k$ such that $f_1(x) = f_2(y) = f_3(x + y) = 1$. Denote by T-Free the set of function triples that are triangle-free, and the distance of a function triple to T-Free is defined as

$$\mathrm{dist}((f_1, f_2, f_3), \text{T-Free}) := \min_{(g_1, g_2, g_3) \in \text{T-Free}} \mathrm{dist}(f_1, g_1) + \mathrm{dist}(f_2, g_2) + \mathrm{dist}(f_3, g_3).$$

As the following reduction and Theorem 2 shows, the multiple-function and single-function case are essentially equivalent.[2]

[1] This is called by Bhattacharyya and Xie [11] the multiple-function case. Green's technique [16] easily generalizes to this case, giving the same bound of tower of twos.

[2] We acknowledge Xie [26] for informing us of the possibility of this reduction.

▶ **Lemma 1** (Xie [26]). *Given any function triple $f_1, f_2, f_3 : \mathbb{F}_2^k \to \{0,1\}$ which is ϵ-far from* T-FREE *and contains N triangles, there is a single function $f : \mathbb{F}_2^{k+2} \to \{0,1\}$ which is $\frac{\epsilon}{4}$-far from triangle-freeness and contains N triangles.*

Proof. Construct f as follows. For each $x \in \mathbb{F}_2^k$, denote by (a, b, x) the $(k+2)$-dimension vector whose last k coordinates are given by x. For each $x \in \mathbb{F}_2^k$, let $f(0,0,x)$ be 0, $f(1,0,x)$ be $f_1(x)$, $f(0,1,x)$ be $f_2(x)$, and $f(1,1,x)$ be $f_3(x)$. It is easy to see that any triangle in f has to have its three "vertices" given by entries from f_1, f_2 and f_3, respectively. The lemma follows immediately. ◀

The *canonical tester* is the naive-looking algorithm that samples $x, y \in \mathbb{F}_2^k$ uniformly at random and returns YES if $f_1(x) = f_2(y) = f_3(x+y) = 1$ and NO otherwise. A tester is said to be *one-sided* if, whenever the input satisfies the property in question, it outputs YES with probability 1. By the following theorem, it is without loss of generality to consider obfuscating the canonical tester.

▶ **Theorem 2** (Bhattacharyya and Xie [11]). *Suppose there is a one-sided tester for* T-FREE *has query complexity $q(\epsilon)$, then the canonical tester has query complexity at most $O(q^2(\epsilon))$. This holds for both the single-function case (when $f_1 = f_2 = f_3$) and the multiple-function case.*

▶ **Definition 3** (Perfect-Matching-Free (PMF) Families of Vectors). Let k and m be integers such that $0 < k < m < 2^k$. A (k, m) perfect-matching-free (PMF) family of vectors is a set of vectors $(a_i, b_i, c_i)_{i=1}^m$, where $a_i, b_i, c_i \in \mathbb{F}_2^k$ and $c_i = a_i + b_i$ for all $i \in [m]$, such that for any permutation triple $\pi_1, \pi_2, \pi_3 \in \mathrm{Sym}([m])$, either $\pi_1 = \pi_2 = \pi_3$, or there exists an $i \in [m]$ such that $a_{\pi_1(i)} + b_{\pi_2(i)} \neq c_{\pi_3(i)}$.

One can permute and then concatenate all a_i's in a (k, m) PMF family and obtains $m!$ vectors in $\mathbb{F}_2^{km}$; the same can be done for b_i's and c_i's. By the property of PMF, each new vector obtained from a_i's forms one and only one triangle with two other vectors obtained from b_i's and c_i's, respecitvely, and they are obtained through exactly the same permutation on $[m]$. This means that to remove all $m!$ triangles in the system, one has to remove at least the same number of vectors. This large ratio between the distance to triangle-freeness and the number of triangles is exactly what is needed to obfuscate a tester. One may go further and take multiple copies of a PMF family and repeats this experiment. An asymptotic calculation would give the following theorem.

▶ **Theorem 4** (Bhattacharyya and Xie [11]). *If (k, m) PMF family of vectors exists, then for small enough ϵ and large enough k, there exists a function triple $f_1, f_2, f_3 : \mathbb{F}_2^k \to \{0,1\}$ that is ϵ-far from triangle-freeness, but the canonical tester needs $\Omega((\frac{1}{\epsilon})^\alpha)$ queries to detect a triangle, where $\alpha = (2 - \frac{\log m}{k})/(1 - \frac{\log m}{k})$.* [3]

Note that the existence of $(k, 2^{k(1-o_k(1))})$ PMF family would imply a super-polynomial lower bound for any one-sided triangle-freeness tester.

The workhorse of our improved lower bound for testing triangle-freeness is the following combinatorial construction. It was implicitly developed by Coppersmith and Winograd [13] for their famed $O(n^{2.376})$-time matrix multiplication algrorithm, and Cohn et al. [12] isolated it and gave it the reinterpretation we use here.

[3] All logarithms in this paper are base 2.

▶ **Definition 5** (Uniquely Solvable Puzzles (USP)). A *uniquely solvable puzzle* (USP) is a set $U \subset \{1,2,3\}^k$ such that, for all permutation triples $\pi_1, \pi_2, \pi_3 \in \mathrm{Sym}(U)$, either $\pi_1 = \pi_2 = \pi_3$, or there exist a $u \in U$ and an index $i \in [k]$ such that at least two of $(\pi_1(u))_i = 1$, $(\pi_2(u))_i = 2$ and $(\pi_3(u))_i = 3$ hold.

A useful way to look at a USP is to think of it as a set of puzzles having three colors, where each color has m pieces. A solution to the puzzle is an arrangement of the pieces into m rows each of size k, such that each row contains one piece of each color, and there is no conflict, i. e., a position occupied by two pieces of different colors. The property in Definition 3 requires that there exists a unique solution to this puzzle, up to permutations on rows.

▶ **Theorem 6** (Coppersmith and Winograd [13], Cohn et al. [12]). *Fixing integer k, the largest USP is of size $\Theta((3/2^{2/3} - o(1))^k)$.*

The upper bound, given by an elegant construction of large USPs in Coppersmith and Winograd's original paper was unfortunately buried in a system of algebraic notations not easy to decipher without a proficiency with that language. For the sake of completeness and to promulgate this beautiful construction, we give its proof, hopefully more accessible, in Appendix A.

3 A Construction of PMF Families via USPs

We now state the main theorem of the paper.

▶ **Theorem 7.** *For any $\epsilon > 0$ and large enough k, there exists a function triple $f_1, f_2, f_3 : \mathbb{F}_2^k \to \{0,1\}$, such that the triple is ϵ-far from being triangle free, and the canonical tester needs $\Omega((\frac{1}{\epsilon})^{13.239})$ queries to detect a triangle in the triple. In addition, any one-sided tester needs $\Omega((\frac{1}{\epsilon})^{6.619})$ queries.*

By Theorem 2 and Theorem 4, Theorem 7 would be an immediate consequence of the following lemma.

▶ **Lemma 8.** *There exists $(k, \Theta((3/2^{2/3} - o(1))^k))$ PMF family of vectors, for all k.*

Proof of Lemma 8. By Theorem 6, it suffices to construct a $(k, |U|)$ PMF family for any USP $U \subset \{1,2,3\}^k$. Let U be $\{u_1, u_2, \ldots, u_m\}$. We construct $3m$ vectors $a_i, b_i, c_i \in \mathbb{F}_2^k$ for $i = 1, 2, \ldots, m$. For each $i \in [m]$, let $a_{i,j}$ be 1 if $u_{i,j} = 1$, and 0 otherwise; let $b_{i,j}$ be 1 if $u_{i,j} = 2$, and 0 otherwise; let $c_{i,j}$ be 1 if $u_{i,j} \neq 3$, and 0 otherwise. It is clear now that $c_i = a_i + b_i$ for all i.

We now show that $\{a_i, b_i, c_i\}_{i=1}^m$ constitutes a PMF family. Note that a naive translation of the property of USP would not give the desired property for PMF: for $\pi_1, \pi_2, \pi_3 \in \mathrm{Sym}([m])$ that are not all equal and such that $u_{\pi_1(i),j} = 1$, $u_{\pi_2(i),j} = 2$ and $u_{\pi_3(i),j} = 3$ for some $i \in [m], j \in [k]$, we will have that $a_{\pi_1(i),j} = b_{\pi_2(i),j} = 1$ and $c_{\pi_3(i),j} = 0$, which does not prevent the sum of a_i and b_i from being c_i in $\mathbb{F}_2^k$. Instead, we observe that for $\pi_1, \pi_2, \pi_3 \in \mathrm{Sym}([m])$ that are not all equal, there must be an $i \in [m]$ and $j \in [k]$ such that $u_{\pi_1(i),j} \neq 1$, $u_{\pi_2(i),j} \neq 2$ and $u_{\pi_3(i),j} \neq 3$: this is because of the conservation of the total number of elements in U. The number of 1's and 2's and 3's in U total at mk, and if, by the property according to Definition 3, under permutations of the puzzles there exist conflicts at some position, then there must be some other position that is not covered by a puzzle of any color. For such i and j we would have $a_{\pi_1(i),j} = b_{\pi_2(i),j} = 0$ and $c_{\pi_3(i),j} = 1$, which means that $a_{\pi_1(i)} + b_{\pi_2(i)} \neq c_{\pi_3(i)}$. This shows that we have indeed constructed a $(k, |U|)$ PMF family. ◀

References

1 Noga Alon. Testing subgraphs in large graphs. *Random Struct. Algorithms*, 21(3-4):359–370, 2002.

2 Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. A combinatorial characterization of the testable graph properties: it's all about regularity. In *STOC*, pages 251–260, 2006.

3 Noga Alon and Asaf Shapira. Testing subgraphs in directed graphs. *J. Comput. Syst. Sci.*, 69(3):354–382, 2004.

4 Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. In *FOCS*, pages 429–438, 2005.

5 Noga Alon, Amir Shpilka, and Christopher Umans. On sunflowers and matrix multiplication. *Computational Complexity*, 22(2):219–243, 2013.

6 Tim Austin and Terence Tao. Testability and repair of hereditary hypergraph properties. *Random Struct. Algorithms*, 36(4):373–463, 2010.

7 Felix A. Behrend. On sets of integers which contain no three terms in arithmetical progression. *Proc. Nat. Acad. Sci.*, 32:331–332, 1946.

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, Elena Grigorescu, Prasad Raghavendra, and Asaf Shapira. Testing odd-cycle-freeness in boolean functions. *Combinatorics, Probability & Computing*, 21(6):835–855, 2012.

10 Arnab Bhattacharyya, Elena Grigorescu, and Asaf Shapira. A unified framework for testing linear-invariant properties. In *FOCS*, pages 478–487, 2010.

11 Arnab Bhattacharyya and Ning Xie. Lower bounds for testing triangle-freeness in boolean functions. In *SODA*, pages 87–98, 2010.

12 Henry Cohn, Robert D. Kleinberg, Balázs Szegedy, and Christopher Umans. Group-theoretic algorithms for matrix multiplication. In *FOCS*, pages 379–388, 2005.

13 Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *J. Symbolic Computation*, 9(3):250–280, 1990.

14 Michael Elkin. An improved construction of progression-free sets. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 886–905, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

15 Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

16 Ben Green. A Szemerédi-type regularity lemma in abelian groups, with applications. *Geom. Funct. Anal.*, 15(2):340–376, 2005.

17 Ishay Haviv and Ning Xie, 2014. Private communication.

18 Tali Kaufman and Madhu Sudan. Algebraic property testing: the role of invariance. In *STOC*, pages 403–412, 2008.

19 Daniel Král, Oriol Serra, and Lluís Vena. On the removal lemma for linear systems over abelian groups. *Eur. J. Comb.*, 34(2):248–259, 2013.

20 Vojtech Rödl and Mathias Schacht. Generalizations of the removal lemma. *Combinatorica*, 29(4):467–501, 2009.

21 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996.

22 R. Salem and D. Spencer. On sets of integers which contain no three in arithmetic progression. *Proc. Nat. Acad. Sci.*, 28:561–563, 1942.

23 Asaf Shapira. Green's conjecture and testing linear-invariant properties. In *STOC*, pages 159–166, 2009.

24 Andrew James Stothers. *On the Complexity of Matrix Multiplication.* PhD thesis, University of Edinburgh, 2010.
25 Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *STOC*, pages 887–898, 2012.
26 Ning Xie, 2010. Private communication.

A Construction of Large Uniquely Solvable Puzzles

In this appendix we present Coppersmith and Winograd (1990)'s construction of large USPs, isolating it from the matrix multiplication context.

The construction makes use of the following theorem:

▶ **Theorem 9** (Salem and Spencer [22]). *Given $\delta > 0$, for all large enough integer M, there is a set $B \subset [M]$ of size $\Omega(M^{1-\delta})$ such that for all $b_i, b_j, b_k \in B$, $b_i + b_j \equiv 2b_k \mod M$ iff $i = j = k$.*

Such constructions of big sets of integers with no arithmetic progressions constitute an important class of combinatorial objects. Improvements over Salem and Spencer's original construction with slightly larger sizes were given by Behrend [7] and Elkin [14], but for our purpose the rougher asymptotic bound of $\Omega(M^{1-\delta})$ suffices.

Now we are ready to describe the construction. We fix a large enough integer N and $M = 2\binom{2N}{N} + 1$. Fix $B \subset [M]$ as given by Theorem 9. Sample $3N$ integers $0 \leq w_j < M$ independently at random for each $j = 0, 1, \cdots, 3N$. We will call these w_j's *weights*. Now consider the set $\mathcal{I}$ of all subsets $I \subset [3N]$ of size N. Let δ_I be the indicator function of subset I, i. e., for each $j \in [3N]$, $\delta_I(j) = 1$ for $j \in I$, and 0 otherwise. The weights we sampled define three mappings from $\mathcal{I}$ to $\mathbb{Z}_M$:

$$\beta_x(I) \equiv \sum_{j=1}^{3N} \delta_I(j) w_j \mod M; \tag{1}$$

$$\beta_y(I) \equiv w_0 + \sum_{j=1}^{3N} \delta_I(j) w_j \mod M; \tag{2}$$

$$\beta_z(I) \equiv \left(w_0 + \sum_{j=1}^{3N} (1 - \delta_I(j)) w_j \right) /2 \mod M. \tag{3}$$

Note that the operation of division by 2 is well defined for β_z, as M is odd.

With these mappings, we will consider each element $b_i \in B$. First, with each $b_i \in B$ we associate all triples (I, J, K), where $I, J, K \in \mathcal{I}$ are pairwise disjoint, and $\beta_x(I) = \beta_y(J) = \beta_z(K) = b_i$. (A triple (I, J, K) is discarded if the members are not pairwise disjoint, or if they are not mapped to be same b_i.) Second, among all triples associated with the same b_i, we arbitrarily remove all but one triple. To construct our USP $U \subset \{1, 2, 3\}^{3N}$, there will be a puzzle u_i for each b_i associated with a nonempty triple (I_i, J_i, K_i), and for each $j \in [3N]$, $u_i(j) = 1$ for $j \in I_i$, $u_i(j) = 2$ for $j \in J_i$, and $u_i(j) = 3$ for $j \in K_i$.

We first check that we indeed obtain a USP family, before going on to prove its expected size.

▶ **Claim 10.** *For any $i_1, i_2, i_3 \in [|B|]$, I_{i_1}, J_{i_2} and K_{i_3} are pairwise disjoint iff $i_1 = i_2 = i_3$.*

Note that Claim 10 suffices for the property of USP (Definition 5).

Proof. Suppose I_{i_1}, J_{i_2} and K_{i_3} are pairwise disjoint, we have that

$$b_{i_1} \equiv \beta_x(i_1) \equiv \sum_{j=1}^{3N} \delta_{I_{i_1}}(j) w_j \mod M; \tag{4}$$

$$b_{i_2} \equiv \beta_y(i_2) \equiv w_0 + \sum_{j=1}^{3N} \delta_{J_{i_2}}(j) w_j \mod M; \tag{5}$$

$$b_{i_3} \equiv \beta_z(i_3) \equiv \left(w_0 + \sum_{j=1}^{3N} (1 - \delta_{K_{i_3}}(j)) w_j \right) /2 \equiv \left(w_0 + \sum_{j=1}^{3N} \delta_{I_{i_1} \cup J_{i_2}} w_j \right) /2 \mod M. \tag{6}$$

Straightforwardly, we will have $b_{i_1} + b_{i_2} - 2b_{i_3} \equiv 0 \mod M$. However, since b_{i_1}, b_{i_2} and b_{i_3} are in B, by the property of B, it can only be that $i_1 = i_2 = i_3$. ◀

We now show that the we indeed have a large USP. This amounts to showing that we have many triples left at the end of the second step of the construction. We first consider the number of triples associated with elements in B in the first step.

▶ **Claim 11.** *Fixing $b_i \in B$, the expected number of triples (I, J, K) associated with b_i in the first step is $\binom{3N}{N,N,N} M^{-2}$.*

Proof. First, by the same calculation as in Claim 10, we know that if two disjoint $I, J \in \mathcal{I}$ are mapped to the same $b_i \in B$ by β_x and β_y, respectively, then their complement, $K = [3N] - (I \cup J)$, must be mapped to be same b_i by β_z. Now there are $\binom{3N}{N,N,N}$ disjoint triples, the probability that each of the two components is mapped to a fixed b_i is M^{-1}, respectively. Moreover, the two events are independent. The claim follows. ◀

▶ **Claim 12.** *Fixing $b_i \in B$, the expected number of triples (I, J, K) associated with b_i that we remove in the second step is at most $\frac{3}{2}\binom{3N}{N,N,N}(\binom{2N}{N} - 1)M^{-3}$.*

Proof. Fixing $b_i \in B$, the expected number of triples (I, J, K) and (I', J', K) $(I \neq I')$ associated with b_i is $\frac{1}{2}\binom{3N}{N,N,N}(\binom{2N}{N} - 1)M^{-3}$. The term $\binom{2N}{N} - 1$ counts the number of I''s disjoint with K and unequal to I. The factor $\frac{1}{2}$ disregards the symmetric case $(I, J, K), (I', J', K)$ and $(I', J', K), (I, J, K)$. The additional M^{-1} here (as compared to the count in Claim 11) indicates the probability of the event $\beta_y(I') = b_i$. Note that this event is independent from the events $\beta_x(I) = b_i$ and $\beta_y(J) = b_i$, even if J' can be equal to I, because of the presence of the weight w_0 in the definition of β_y. Repeat the argument for the cases when two triples coincide on the first or second subset, and the claim follows. (The event that two triples associated with the same b_i disagree on each subset they contain is neglected here, since its probability is significantly smaller than that of the case analyzed here. For large N and M this is easily accommodated.) ◀

Therefore, by our choice of M, the expected number of triples associated with each b_i remaining after the second step is at least

$$\binom{3N}{N,N,N} M^{-2} - \frac{3}{2}\binom{3N}{N,N,N}\left(\binom{2N}{N} - 1\right) M^{-3} \geq \frac{1}{4}\binom{3N}{N,N,N} M^{-2}.$$

With a standard probabilistic argument, we conclude that there exists a choice of w_j's such that the size of USP we obtain is at least

$$\frac{1}{4}\binom{3N}{N,N,N} M^{-2}|B| = \frac{1}{4}\binom{3N}{N,N,N} M^{-2} M^{1-\delta}.$$

Substituting our choice of M and applying the Stirling's formula, we get the desired asymptotic bound of $(3/2^{2/3} - o(1))^{3N}$ for the size of USP.

Ferromagnetic Potts Model: Refined #BIS-hardness and Related Results*

Andreas Galanis[†1], Daniel Štefankovič[‡2], Eric Vigoda[§3], and Linji Yang[4]

1 **University of Oxford, Wolfson Building, Parks Road, Oxford, OX1 3QD, UK**
andreas.galanis@cs.ox.ac.uk
2 **University of Rochester, Rochester, NY, 14627, USA**
stefanko@cs.rochester.edu
3 **Georgia Institute of Technology, Atlanta, GA, 30332, USA**
vigoda@cc.gatech.edu
4 **Facebook, Inc., Menlo Park, CA, 94025, USA**
ljyang@gatech.edu

Abstract

Recent results establish for the hard-core model (and more generally for 2-spin antiferromagnetic systems) that the computational complexity of approximating the partition function on graphs of maximum degree Δ undergoes a phase transition that coincides with the uniqueness/non-uniqueness phase transition on the infinite Δ-regular tree. For the ferromagnetic Potts model we investigate whether analogous hardness results hold. Goldberg and Jerrum showed that approximating the partition function of the ferromagnetic Potts model is at least as hard as approximating the number of independent sets in bipartite graphs, so-called #BIS-hardness. We improve this hardness result by establishing it for bipartite graphs of maximum degree Δ. To this end, we first present a detailed picture for the phase diagram for the infinite Δ-regular tree, giving a refined picture of its first-order phase transition and establishing the critical temperature for the coexistence of the disordered and ordered phases. We then prove for all temperatures below this critical temperature (corresponding to the region where the ordered phase "dominates") that it is #BIS-hard to approximate the partition function on bipartite graphs of maximum degree Δ.

The #BIS-hardness result uses random bipartite regular graphs as a gadget in the reduction. The analysis of these random graphs relies on recent results establishing connections between the maxima of the expectation of their partition function, attractive fixpoints of the associated tree recursions, and induced matrix norms. In this paper we extend these connections to random regular graphs for all ferromagnetic models. Using these connections, we establish the Bethe prediction for every ferromagnetic spin system on random regular graphs, which says roughly that the expectation of the log of the partition function Z is the same as the log of the expectation of Z. As a further consequence of our results, we prove for the ferromagnetic Potts model that the Swendsen-Wang algorithm is torpidly mixing (i. e., exponentially slow convergence to its stationary distribution) on random Δ-regular graphs at the critical temperature for sufficiently large q.

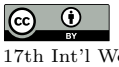
1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Ferromagnetic Potts model, approximate counting, spin systems, phase transition, random regular graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.677

* Full version [15].

† 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.
‡ Research supported in part by NSF grant CCF-1318374.
§ Research supported in part by NSF grant CCF-1217458.



17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 677–691

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

1 Background

1.1 Spin Systems

We study the ferromagnetic Potts model and present tools which are useful for any ferromagnetic spin system on random regular graphs. Hence we begin with a general definition of a spin system.

A spin system is defined, for an n-vertex graph $G = (V, E)$ and integer $q \geq 2$, on the space Ω of configurations σ which are assignments $\sigma : V \to [q]$. The model is characterized by its energy or Hamiltonian $H(\sigma)$ which is a function of the spin assignments to the vertices. In the classical examples of the Ising ($q = 2$) and Potts ($q \geq 3$) models without external field, the Hamiltonian $H(\sigma)$ is the number of monochromatic edges in σ. Each configuration has a weight $w(\sigma) = \exp(-\beta H(\sigma))$ for a parameter β corresponding to the "inverse temperature" which controls the strength of edge interactions.

The *Gibbs distribution* is defined as $\mu(\sigma) = w(\sigma)/Z$ where $Z = Z_G(\mathbf{B}) = \sum_\sigma w(\sigma)$ is the *partition function*. In our general setup, a *specification* of a q-state spin model is defined by a symmetric $q \times q$ interaction matrix $\mathbf{B}$ with non-negative entries. The weight of a configuration in this general setup is given by:

$$w(\sigma) = \prod_{\{u,v\} \in E} B_{\sigma(u),\sigma(v)}.$$

Many of our results also apply to models with arbitrary external fields since we will work with Δ-regular graphs and in this case the external field can be incorporated into the interaction matrix.

The Ising ($q = 2$) and Potts ($q > 2$) models have interaction matrices with diagonal entries $B := \exp(-\beta)$ and off-diagonal entries 1. The models are called ferromagnetic if $B > 1$ since then neighboring spins prefer to align and antiferromagnetic if $B < 1$. The hard-core model is an example of a 2-spin antiferromagnetic system, its interaction matrix is defined so that Ω is the set of independent sets of G and configuration $\sigma \in \Omega$ has weight $w(\sigma) = \lambda^{|\sigma|}$ for activity $\lambda > 0$.

We are not aware of a general definition of ferromagnetic and antiferromagnetic models. We use the following notions which generalize the analogous notions for 2-spin and for the Potts model. The ferromagnetic definition captures that neighboring spins preferring to align (see [15, Observation 1] in the full version of this paper). To avoid degenerate cases, we assume throughout this paper that $\mathbf{B}$ is ergodic, that is, irreducible and aperiodic, see [15, Section 1.2] in the full version for a detailed discussion. Hence, by the Perron-Frobenius theorem (and since $\mathbf{B}$ is non-negative) the eigenvalue of $\mathbf{B}$ with the largest magnitude is positive.

▶ **Definition 1.** A model is called **ferromagnetic** if $\mathbf{B}$ is positive definite. Equivalently we have that all of its eigenvalues are positive and also that

$$\mathbf{B} = \hat{\mathbf{B}}^T \hat{\mathbf{B}}, \tag{1}$$

for some $q \times q$ matrix $\hat{\mathbf{B}}$.

In contrast to the above notion of a ferromagnetic system, in [17] a model is called **antiferromagnetic** if all of the eigenvalues of $\mathbf{B}$ are negative except for the largest (which, as noted above, is positive).

1.2 Known Connections to Phase Transitions

Exact computation of the partition function is #P-complete, even for very restricted classes of graphs [23]. Hence we focus on whether there is a fully-polynomial (randomized or deterministic) approximation scheme, a so-called FPRAS or FPTAS.

One of our goals in this paper is to refine our understanding of connections between approximating the partition function on graphs of maximum degree Δ with phase transitions on the infinite Δ-regular tree $\mathbb{T}_\Delta$. A phase transition of particular interest in the infinite tree $\mathbb{T}_\Delta$ is the uniqueness/non-uniqueness threshold. Roughly speaking, in the uniqueness phase, if one fixes a so-called "boundary condition" which is a configuration σ_ℓ (for instance, an independent set in the hard-core model) on the vertices distance ℓ from the root, then in the Gibbs distribution conditioned on this configuration, is the root "unbiased"? Specifically, for all sequences (σ_ℓ) of boundary conditions, in the limit $\ell \to \infty$, does the root have the same marginal distribution? If so, there is a unique Gibbs measure on the infinite tree and hence we say the model is in the uniqueness region. If there are sequences of boundary conditions which influence the root in the limit then we say the model is in the non-uniqueness region.

For 2-spin antiferromagnetic spin systems, it was shown that there is an FPTAS for estimating the partition function for graphs of maximum degree Δ when the infinite tree $\mathbb{T}_\Delta$ is in the uniqueness phase [28]. On the other side, unless NP=RP, there is no FPRAS for the partition function for Δ-regular graphs when $\mathbb{T}_\Delta$ is in the non-uniqueness phase [36] (see also [16]). Recently, an analogous NP-hardness result was shown for approximating the number of k-colorings on triangle-free Δ-regular graph for even k when $k < \Delta$. In contrast to the above inapproximability results for antiferromagnetic systems, for the Ising model with or without external field [26] and for 2-spin ferromagnetic spin systems without external field [22] there is an FPRAS for all graphs. The situation for ferromagnetic multi-spin models, the ferromagnetic Potts being the most prominent example, is more intricate.

#BIS refers to the problem of computing the number of independent sets in bipartite graphs. A series of results has presented evidence that there is unlikely to be a polynomial-time algorithm for #BIS, since a number of unsolved counting problems have be shown to be #BIS-hard (for example, see [13, 2, 7]). The growing anecdotal evidence for #BIS-hardness suggests that the problem is intractable, though weaker than NP-hardness. More recently it was shown in [6] that for antiferromagnetic 2-spin models it is #BIS-hard to approximate the partition function on bipartite graphs of maximum degree Δ when the parameters of the model lie in the non-uniqueness region of the infinite Δ-regular tree $\mathbb{T}_\Delta$.

2 Results for the Potts Model

2.1 #BIS-hardness for the Potts model

Goldberg and Jerrum [20] showed that approximating the partition function of the ferromagnetic Potts model is #BIS-hard, hence it appears likely that the ferromagnetic Potts model is inapproximable for general graphs. We refine this #BIS-hardness result for the ferromagnetic Potts model. We prove that approximating the partition function for the ferromagnetic Potts model on *bipartite* graphs of maximum degree Δ is #BIS-hard for temperatures above the appropriate phase transition point in the infinite tree $\mathbb{T}_\Delta$. The appropriate phase transition in the Potts model is not the uniqueness/non-uniqueness threshold, but rather it is the ordered/disordered phase transition which occurs at $B = \mathfrak{B}_o$ as explained in the next section.

Formally, we study the following problem.

Name. #BipFerroPotts(q, B, Δ).
Instance. A bipartite graph G with maximum degree Δ.
Output. The partition function for the q-state Potts model on G.

We use the notion of approximation-preserving reductions, denoted as $\leq_{AP}$, formally defined in [13]. We can now formally state our main result.

▶ **Theorem 2.** *For all $q \geq 3$, all $\Delta \geq 3$, for the ferromagnetic q-state Potts model, for any $B > \mathfrak{B}_o$,*

$$\#\text{BIS} \leq_{AP} \#\text{BipFerroPotts}(q, B, \Delta),$$

where $\mathfrak{B}_o$ is given by (4)*.*

2.2 Potts Model Phase Diagram

To understand the critical point $\mathfrak{B}_o$ we need to delve into the nature of the phase transition in the ferromagnetic Potts model on the infinite Δ-regular tree $\mathbb{T}_\Delta$. We focus on how the phase transition manifests on a random Δ-regular graph.

For a configuration $\sigma \in \Omega$, denote the set of vertices assigned spin i by $\sigma^{-1}(i)$. Let $\triangle_q$ denote the $(q-1)$-simplex:

$$\triangle_t = \{(x_1, x_2, \ldots, x_t) \in \mathbb{R}^t \,|\, \textstyle\sum_{i=1}^t x_i = 1 \text{ and } x_i \geq 0 \text{ for } i = 1, \ldots, t\}.$$

We refer to $\boldsymbol{\alpha} \in \triangle_q$ as a *phase*. For a phase $\boldsymbol{\alpha}$, denote the set of configurations with frequencies of colors given by $\boldsymbol{\alpha}$ as:

$$\Sigma^{\boldsymbol{\alpha}} = \left\{\sigma : V \to \{1, \ldots, q\} \,\middle|\, |\sigma^{-1}(i)| = \lfloor \alpha_i n \rfloor \text{ for } i = 1, \ldots, q\right\},$$

and denote the partition function restricted to these configurations by:

$$Z_G^{\boldsymbol{\alpha}} = \textstyle\sum_{\sigma \in \Sigma^{\boldsymbol{\alpha}}} w_G(\sigma).$$

Let $\mathcal{G}$ denote the uniform distribution over Δ-regular graphs. Denote the exponent of the first moment as:

$$\Psi_1(\boldsymbol{\alpha}) := \Psi_1^{\mathbf{B}}(\boldsymbol{\alpha}) := \lim_{n \to \infty} \frac{1}{n} \log \mathbf{E}_{\mathcal{G}}\left[Z_G^{\boldsymbol{\alpha}}\right]. \tag{2}$$

The expression for Ψ_1 can be found in the full version of this paper, see [15, Section 4].

Those $\boldsymbol{\alpha}$ which are global maxima of Ψ_1 we refer to as *dominant* phases. We will see in Section 3.2 that the candidates for dominant phases correspond to stable fixpoints of the so-called tree recursions. There will be two phases of particular interest; we refer to these phases as the disordered phase and the ordered phase. The disordered phase is the uniform vector $\boldsymbol{\alpha} = (1/q, \ldots, 1/q)$. The ordered phase refers to a phase with one color dominating in the following sense: one coordinate is equal to $a > 1/q$ and the other $q-1$ coordinates are equal to $(1-a)/(q-1)$. Due to the symmetry of the Potts model, when the ordered phase dominates, in fact, the q symmetric ordered phases dominate. These ordered phases have a specific $a = a(B)$ which corresponds to a fixpoint of the tree recursions. The exact definition of this marginal a is not important at this stage, and hence we defer its explicit definition to a more detailed discussion which can be found in the full version of this paper, see [15, Section 8].

One of the difficulties for the Potts model is that the nature of the uniqueness/non-uniqueness phase transition on $\mathbb{T}_\Delta$ is inherently different from that of the Ising model. The

ferromagnetic Ising model undergoes a second-order phase transition on $\mathbb{T}_\Delta$ which manifests itself on random Δ-regular graphs in the following manner. In the uniqueness region the disordered phase dominates, and in the non-uniqueness region the 2 ordered phases dominate.

In contrast, the ferromagnetic Potts model undergoes a first-order phase transition at the critical activity $\mathfrak{B}_u$. For $B < \mathfrak{B}_u$ there is a unique Gibbs measure on $\mathbb{T}_\Delta$. For $B \geq \mathfrak{B}_u$ there are multiple Gibbs measures on $\mathbb{T}_\Delta$, however there is a second critical activity $\mathfrak{B}_o$ corresponding to the disordered/ordered phase transition: for $B \leq \mathfrak{B}_o$ the disordered phase dominates, and for $B \geq \mathfrak{B}_o$ the ordered phases dominate (and at the critical point $\mathfrak{B}_o$ all of these $q+1$ phases dominate).

We present a detailed picture of the phase diagram for the ferromagnetic Potts model. Previously, Häggström [24] established the uniqueness threshold $\mathfrak{B}_u$ by studying percolation in the random cluster representation. In addition, Dembo et al. [11, 12] studied the ferromagnetic Potts model (including the case with an external field) and proved that for $B > \mathfrak{B}_u$, either the disordered or the q ordered phases are dominant, but they did not establish the precise regions where each phase dominates. For the simpler case of the complete graph (known as the Curie-Weiss model), [9] detailed the phase diagram.

Häggström [24] established that the uniqueness/non-uniqueness threshold for the infinite tree $\mathbb{T}_\Delta$ occurs at $\mathfrak{B}_u$ which is the unique value of B for which the following polynomial has a double root in $(0,1)$:

$$(q-1)x^\Delta + (2-B-q)\,x^{\Delta-1} + Bx - 1. \tag{3}$$

The disordered phase is dominant in the uniqueness region and continues to dominate until the following activity (which was considered by Peruggi et al. [33]):

$$\mathfrak{B}_o := \frac{q-2}{(q-1)^{(1-2/\Delta)}-1}. \tag{4}$$

Finally, Häggström [24] considers the following activity $\mathfrak{B}_{rc}$, which he conjectures is a (second) threshold for uniqueness of the random-cluster model, defined as:

$$\mathfrak{B}_{rc} := 1 + \frac{q}{\Delta-2}.$$

Note, $\mathfrak{B}_u < \mathfrak{B}_o < \mathfrak{B}_{rc}$.

We prove the following picture for the phase diagram for the ferromagnetic Potts model (the proof can be found in the full version [15, Section 8]). Note, to prove that a function has a local maximum at a critical point, a standard approach is to show that its Hessian is negative definite. We often need this stronger condition in our proofs, hence we call such a critical point a *Hessian local maximum*. Moreover, those dominant phases $\boldsymbol{\alpha}$ where the Hessian of Ψ_1 is negative definite are called *Hessian dominant* phases. Note that dominant phases always exist but a dominant phase can fail to be Hessian (when some eigenvalue of the underlying Hessian is equal to zero). In Section 3.2, we give an alternative formulation of the Hessian condition in terms of the local stability of fixpoints of the tree recursions.

▶ **Theorem 3.** *For all $q \geq 3$ and $\Delta \geq 3$, for the ferromagnetic Potts model the following holds at activity B:*

- *$B < \mathfrak{B}_u$: There is a unique infinite-volume Gibbs measure on $\mathbb{T}_\Delta$. The disordered phase is Hessian dominant phase, and there are no other local maxima of Ψ_1.*
- *$\mathfrak{B}_u < B < \mathfrak{B}_{rc}$: The local maxima of Ψ_1 are the disordered phase $\mathbf{u}$ and the q ordered phases (the ordered phases are permutations of each other). All of these $q+1$ phases are Hessian local maxima. Moreover:*

$\mathfrak{B}_u < B < \mathfrak{B}_o$: *The disordered phase is Hessian dominant.*
$B = \mathfrak{B}_o$: *Both the disordered phase and the ordered phases are Hessian dominant.*
$\mathfrak{B}_o < B < \mathfrak{B}_{rc}$: *The ordered phases are Hessian dominant.*

$B \geq \mathfrak{B}_{rc}$: *The q ordered phases (which are permutations of each other) are Hessian dominant. For $B > \mathfrak{B}_{rc}$ there are no other local maxima of Ψ_1.*

2.3 Swendsen-Wang Algorithm

An algorithm of particular interest for the ferromagnetic Potts model is the Swendsen-Wang algorithm. The Swendsen-Wang algorithm is an ergodic Markov chain whose stationarity distribution is the Gibbs distribution. It utilizes the random-cluster representation to overcome potential "bottlenecks" for rapid mixing that are expected to arise in the non-uniqueness region. As a consequence of the above picture for the phase diagram on the infinite tree $\mathbb{T}_\Delta$ and our tools for analyzing random regular graphs, we can prove torpid mixing of the Swendsen-Wang algorithm at activities near the disordered/ordered phase transition point $\mathfrak{B}_o$. (Torpid mixing means that the mixing time is exponentially slow.)

The Swendsen-Wang algorithm utilizes the random cluster representation of the Potts model to potentially overcome bottlenecks that obstruct the simpler Glauber dynamics. It is formally defined as follows. From a configuration $X_t \in \Omega$:

- Let M be the set of monochromatic edges in X_t.
- For each edge $e \in M$, delete it with probability $1/B$. Let M' denote the set of monochromatic edges that were not deleted.
- In the graph (V, M'), for each connected component, choose a color uniformly at random from $[q]$ and assign all vertices in that component the chosen color. Let X_{t+1} denote the resulting spin configuration.

There are few results establishing rapid mixing of the Swendsen-Wang algorithm beyond what is known for the Glauber dynamics, see [37] for recent progress showing rapid mixing on the 2-dimensional lattice. However, there are several results establishing torpid mixing of the Swendsen-Wang algorithm at a critical value for the q-state ferromagnetic Potts model: on the complete graph ($q \geq 3$) [21], on Erdös-Rényi random graphs ($q \geq 3$) [8], and on the d-dimensional integer lattice $\mathbb{Z}^d$ (q sufficiently large) [3, 4].

Using our detailed picture of the phase diagram of the ferromagnetic Potts model and our generic second moment analysis for ferromagnetic models on random regular graphs which we explain in a moment, we establish torpid mixing on random Δ-regular graphs at the phase coexistence point $\mathfrak{B}_o$.

▶ **Theorem 4.** *For all $\Delta \geq 3$ and $q \geq 2\Delta/\log\Delta$, with probability $1-o(1)$ over the choice of a random Δ-regular graph, for the ferromagnetic Potts model with $B = \mathfrak{B}_o$, the Swendsen-Wang algorithm has mixing time* $\exp(\Omega(n))$.

3 Results for Ferromagnetic Models

3.1 Second Moment and Bethe Prediction Results

We analyze the Gibbs distribution on random Δ-regular graphs using second moment arguments. The challenging aspect of the second moment is determining the phase that dominates, as we will describe more precisely momentarily. In a straightforward analysis of the second moment, this reduces to an optimization problem over q^4 variables for a

complicated expression. Even for $q = 2$ tackling this requires significant effort (see, for example, [32] for the hard-core model).

In a recent paper [17] we analyzed antiferromagnetic systems on *bipartite* random Δ-regular graphs, to use as gadgets for inapproximability results. In that work we presented a new approach for simplifying the analysis of the second moment for antiferromagnetic models using the theory of matrix norms. In this paper we extend that approach using the theory of matrix norms to analyze the second moment for random Δ-regular graphs (non-bipartite) for ferromagnetic systems. We obtain a short, elegant proof that the exponential order of the second moment is twice the exponential order of the first moment.

Denote the leading term of the second moment as

$$\Psi_2(\boldsymbol{\alpha}) := \Psi_2^{\mathbf{B}}(\boldsymbol{\alpha}) := \lim_{n\to\infty} \frac{1}{n} \log \mathbf{E}_{\mathcal{G}}\left[(Z_G^{\boldsymbol{\alpha}})^2 \right]. \tag{5}$$

Our main technical result is the analysis of the second moment. In particular, we will relate the maximum of the second moment to the maximum of the first moment. To analyze the second moment we need to determine the phase $\boldsymbol{\alpha}$ that maximizes Ψ_2. We first show how to reexpress the maximum of Ψ_1 in a form that can be readily expressed in terms of matrix norms. The details are given in [15, Section 5.1] of the full version of this paper. Then, using the Cholesky decomposition of the interaction matrix $\mathbf{B}$ and properties of matrix norms we show that the maximum of Ψ_2 equals the value of a function at a tensor product of the dominant phases of the first moment. From there, we obtain the following theorem, whose proof can be found in [15, Section 5.2] of the full version.

▶ **Theorem 5.** *For a ferromagnetic model with interaction matrix* $\mathbf{B}$*,*

$$\max_{\boldsymbol{\alpha}} \Psi_2(\boldsymbol{\alpha}) = 2 \max_{\boldsymbol{\alpha}} \Psi_1(\boldsymbol{\alpha}).$$

In particular, for dominant $\boldsymbol{\alpha}$*,* $\Psi_2(\boldsymbol{\alpha}) = 2\Psi_1(\boldsymbol{\alpha})$*.*

Combining Theorem 5 with an elaborate variance analysis known as the small subgraph conditioning method allows us to prove concentration for $Z_G^{\boldsymbol{\alpha}}$ (see Lemma 10). In particular, we verify the so-called *Bethe prediction* for general ferromagnetic models on random Δ-regular graphs, which is captured in our setting by equation (6) in the following theorem.

▶ **Theorem 6.** *Let* $\mathbf{B}$ *specify a ferromagnetic model. Then, if there exists a Hessian dominant phase, it holds that*

$$\lim_{n\to\infty} \frac{1}{n} \mathbf{E}_{\mathcal{G}}[\log Z_G] = \lim_{n\to\infty} \frac{1}{n} \log \mathbf{E}_{\mathcal{G}}[Z_G]. \tag{6}$$

Note that for a ferromagnetic model the interaction matrix $\mathbf{B}$ is positive definite and hence the entries on the diagonal are all positive. Thus Z_G is always positive for every graph G.

Theorem 6 can be extended to general models (not necessarily ferromagnetic) on random Δ-regular graphs under the stronger assumption that there is a unique semi-translation invariant Gibbs measure on $\mathbb{T}_\Delta$. In this setting, one also obtains the analogue of Theorem 5 and as a consequence concentration for $Z_G^{\boldsymbol{\alpha}}$ for the (unique) dominant phase $\boldsymbol{\alpha}$, which can be used to verify (in complete analogy) equation (6), see [15, Section 11.2] in the full version for details and a more thorough discussion.

3.2 Connection to Tree Recursions

As a consequence of Theorem 5, to analyze ferromagnetic models on random regular graphs, one only needs to analyze the first moment. To simplify the analysis of the first moment, we establish the following connection to the so-called tree recursions. An analogous connection was established in [17] for antiferromagnetic models on random bipartite Δ-regular graphs.

A key concept are the following recursions corresponding to the partition function on trees, and hence we refer to them as the (depth one) *tree recursions*:

$$\widehat{R}_i \propto \Big(\sum_{j=1}^{q} B_{ij} R_j\Big)^{\Delta-1} \tag{7}$$

The fixpoints of the tree recursions are those $\mathbf{R} = (R_1, \ldots, R_q)$ such that: $\widehat{R}_i \propto R_i$ for all $i \in [q]$. We refer to a fixpoint $\mathbf{R}$ of the tree recursions as *Jacobian attractive* if the Jacobian at $\mathbf{R}$ has spectral radius less than 1. We prove the following theorem detailing the connections between the tree recursions and the critical points of the partition function for random regular graphs.

▶ **Theorem 7.** *Assume that the model is ferromagnetic. Jacobian attractive fixpoints of the (depth one) tree recursions are in one-to-one correspondence with the Hessian local maxima of Ψ_1.*

The above connection fails for antiferromagnetic models, e.g., for the antiferromagnetic Potts model the uniform fixpoint is a global maximum but it is not a stable fixpoint of the tree recursions for small enough temperature. (In fact, for antiferromagnetic models every solution of the tree recursions is a local maximum, see [15, Remark 3] in the full version.)

Using the above connection we establish the detailed picture for the dominant phases of the ferromagnetic Potts model as stated in Theorem 3.

3.3 Organization

In the following section we prove Theorem 2 showing #BIS-hardness for the Potts model in the ordered region. We also give the proof of Theorem 6 for the Bethe prediction in ferromagnetic models on random Δ-regular graphs in Section 5. The proofs of Theorems 3, 4, 5, 7 are given in the full version of this paper [15]. Specifically, in [15, Section 5] we analyze the second moment and thereby prove Theorem 5 for ferromagnetic models. In [15, Section 10] we prove Theorem 4 establishing torpid mixing of the Swendsen-Wang algorithm at the critical value $B = \mathfrak{B}_o$. We prove the connection between Jacobian attractive fixpoints of the tree recursions and the Hessian local maxima of Ψ_1 in [15, Section 6] and hence obtain Theorem 7. We then use this connection to prove Theorem 3 detailing the phase diagram in [15, Section 8].

4 #BIS-hardness for Potts

We first give a rough description of our reduction. We will construct a gadget G which is a balanced, bipartite graph on $(2 + o(1))n$ vertices. There will be $m' = O(n^{1/8})$ vertices on each side of G which will have degree $\Delta - 1$, the remainder have degree Δ. The key is that G behaves similarly to a random bipartite Δ-regular graph. Hence, the q ordered phases will dominate (for B above $\mathfrak{B}_o$). We will take an instance H for #FerroPotts(q,B) where H has m' vertices. We then replace each vertex in H by a gadget G. Then we will

use the degree $\Delta-1$ vertices in these gadgets to encode the edges of H, while preserving bipartiteness. The resulting graph H^G will have bounded degree Δ and the Potts model on H^G will "simulate" the Potts model on H.

The gadget G is defined by two parameters θ, ψ where $0 < \theta, \psi < 1/8$. The gadget is identical to that used by Sly [35]. The construction of the gadget G has two parts. First construct the following bipartite graph $\overline{G}$ with vertex set $V^+ \cup V^-$. For $s \in \{+,-\}$, $|V^s| = n + m'$ where m' will be defined precisely later. Take Δ random perfect matchings between V^+ and V^-. Then remove a matching of size m' from one of the Δ matchings. Call this graph $\overline{G}$. In the second stage, for each side of $\overline{G}$, partition the degree $\Delta-1$ vertices into n^θ equal sized sets and attach to each set a $(\Delta-1)$-ary tree of depth ℓ where $\ell = \lfloor \psi \log_{\Delta-1} n \rfloor$. (Use the vertices of $\overline{G}$ as the leaves of these trees.) Hence each side contains n^θ trees of size n^ψ. (More precisely, $(\Delta-1)^{\lfloor \theta \log_{\Delta-1} n \rfloor}$ trees of size $(\Delta-1)^{\lfloor \psi \log_{\Delta-1} n \rfloor}$.) This defines the gadget G. For $s \in \{+,-\}$, let R^s denote the roots of the trees on side s. Notice that the roots R^s have degree $\Delta-1$ and these will be used to encode the edges of H as described above. Note that $m' = (\Delta-1)^{\lfloor \theta \log_{\Delta-1} n \rfloor + \lfloor \psi \log_{\Delta-1} n \rfloor}$ and $m' = O(n^{1/8})$. Finally, let $U^+ \cup U^-$ denote the vertices of degree Δ in the initial graph $\overline{G}$ and $W^+ \cup W^-$ denote the vertices of degree $\Delta-1$ in $\overline{G}$.

Denote by $G = (V, E)$ the final graph. Recall, for a configuration $\sigma \in \Omega$, the set of vertices assigned spin i is denoted by $\sigma^{-1}(i)$. The phase of a configuration $\sigma : V \to [q]$ is defined as the dominant spin among vertices in $U = U^+ \cup U^-$:

$$Y(\sigma) := \arg\max_{i \in [q]} |\sigma^{-1}(i) \cap U|,$$

where ties are broken with an arbitrary deterministic criterion (e.g., the lowest index).

The gadget G behaves like a random bipartite Δ-regular graph because $m' \ll n$, as we will detail in the upcoming Lemma 8. Hence, since $B > \mathfrak{B}_o$, Theorem 3 implies that the q ordered phases are dominant. Therefore, we will get that for a sample σ from the Gibbs distribution, the phase of σ will be (close to) uniformly distributed over these q ordered phases. Let phase i refer to the ordered phase where spin i is the majority. Once we condition on the phase for the vertices in U, say it is phase i, then each of the roots, roughly independently, will have spin i with probability $\approx p$ and spin $j \neq i$ with probability $\approx (1-p)/(q-1)$ where p is the probability that the root of the infinite $(\Delta-1)$-ary tree has spin i in the ordered phase i. [1] Hence, for each of the q possible phases, we define the following product distribution on the configurations $\sigma_R : R \to [q]$. For $i \in [q]$, let

$$Q_R^i(\sigma_R) = p^{|\sigma_R^{-1}(i)|} \Big(\frac{1-p}{q-1}\Big)^{|R \setminus \sigma_R^{-1}(i)|}.$$

The following lemma gives the precise formulation of the aforementioned properties of the gadget and is proved using methods in [35]. The proof is given in [15, Section 9.1] of the full version.

▶ **Lemma 8.** *For every $q, \Delta \geq 3$ and $B > \mathfrak{B}_o$, there exist constants $\theta, \psi > 0$ such that the graph G satisfies the following with probability $1 - o(1)$ over the choice of the graph:*

[1] The ordered phase $\boldsymbol{\alpha} = (a, (1-a)/(q-1), \ldots, (1-a)/(q-1))$ specifies the marginal probabilities for the root of the infinite Δ-regular tree. To account for the root having degree $\Delta-1$ one obtains that:

$$p = \frac{a^{(\Delta-1)/\Delta}}{(a/(1-a))^{(\Delta-1)/\Delta} + (q-1)^{1/\Delta}}.$$

1. *The phases occur with roughly equal probability, so that for every phase $i \in [q]$, we have*

$$\left|\mu_G\big(Y(\sigma) = i\big) - \frac{1}{q}\right| \leq n^{-2\theta}.$$

2. *Conditioned on the phase i, the spins of vertices in R are approximately independent, that is,*

$$\max_{\sigma_R} \left|\frac{\mu_G\big(\sigma_R \,|\, Y = i\big)}{Q_R^i(\sigma_R)} - 1\right| \leq n^{-2\theta}.$$

With Lemma 8 at hand, we can now formally state the reduction that we sketched earlier. Let $B > \mathfrak{B}_o$. Let H be a graph on n' vertices, where $n' \leq n^{\theta/4}$ and θ is as in Lemma 8. Assuming an FPRAS for the ferromagnetic Potts model on max degree Δ graphs and temperature B, we will show that we can approximate $Z_H(B^*)$, the partition function of H in the ferromagnetic Potts model with temperature B^*, where B^* will be determined shortly.

To do this, we first construct a graph H^G. First, take $|H|$ disconnected copies of the gadget G in Lemma 8 and identify each copy with a vertex $v \in H$. Denote by $\hat{H}^G$ the resulting graph, G_v the copy of the gadget associated to the vertex v in H and by R_v^+, R_v^-, R_v the images of R^+, R^-, R in the gadget G_v, respectively. Finally, we denote by R_H the set of vertices $\cup_v R_v$. We next add the edges of H in $\hat{H}^G$. To do this, fix an arbitrary orientation of the edges of H. For each oriented edge (u, v) of H, we add an edge between one vertex in R_u^+ and one vertex in R_v^-, using mutually distinct vertices for distinct edges of H. The resulting graph will be denoted by H^G. Note that H^G is bipartite and has maximum degree Δ.

For a graph H and activity $B \geq 1$, recall that $Z_H(B)$ is the partition function for the ferromagnetic Potts model at activity B on the graph H. We have the following connection:

▶ **Lemma 9.** *Let $\Delta, q \geq 3$ and $B > \mathfrak{B}_o$. There exists B^* such that the following holds*

$$\big(1 - O(n^{-\theta})\big)\frac{q^{n'} Z_{H^G}(B)}{C_H\big(Z_G(B)\big)^{n'}} \leq Z_H(B^*) \leq \big(1 + O(n^{-\theta})\big)\frac{q^{n'} Z_{H^G}(B)}{C_H\big(Z_G(B)\big)^{n'}},$$

where $C_H = D^{|E(H)|}$ and $D = 1 + (B-1)\Big(\frac{2p(1-p)}{(q-1)^2} + (q-2)\frac{(1-p)^2}{(q-1)^2}\Big)$.

Using Lemma 9 we can now prove that for all $\Delta \geq 3$, all $B > \mathfrak{B}_o$, it is #BIS-hard to approximate the partition function for the ferromagnetic Potts model on bipartite graphs of maximum degree Δ.

Proof of Theorem 2. Goldberg and Jerrum [20] showed that for every B it is #BIS-hard to approximate the partition function of the ferromagnetic Potts on all graphs. Fix $\Delta, q \geq 3$ and $B > \mathfrak{B}_o$ for which we intend to prove Theorem 2. Let B^* be defined by Lemma 9. We first show that an FPRAS for approximating the partition function with activity B on graphs with maximum degree Δ implies an FPRAS for approximating the partition function with activity B^* on all graphs. It will then be clear that our reduction is in fact approximation-preserving and hence the theorem will be proven.

Suppose that there exists an FPRAS for approximating the partition function with activity B on graphs with maximum degree Δ. Take an input instance H for which we would like to estimate the partition function of the Potts model at activity B^*. First generate a random gadget G using the construction defined earlier. This graph G satisfies the properties in Lemma 8 with probability $1 - o(1)$. Approximate the partition function

of G at activity B within a multiplicative factor $1 \pm \varepsilon/10n'$ using our presumed FPRAS. Also, using the presumed FPRAS approximate the partition function of H^G at activity B within a multiplicative factor $1 \pm \varepsilon/2$. The bounds for $Z_H(B^*)$ in Lemma 9 are then within a factor $1 \pm \varepsilon$ for sufficiently large n, giving an FPRAS for approximating the partition function at activity B^*. This, together with the result of [20], implies an FPRAS for counting independent sets in bipartite graphs. ◀

Proof of Lemma 9. Recall that $\hat{H}^G$ are the disconnected copies of the gadgets, as defined in the construction of H^G. Note, $Z_{\hat{H}^G}(B) = \big(Z_G(B)\big)^{n'}$. Hence to prove the lemma it suffices to analyze $\frac{Z_{H^G}(B)}{Z_{\hat{H}^G}(B)}$.

For a configuration σ on H^G, for each $v \in H$, let $Y_v(\sigma)$ denote the phase of σ on G_v. Denote the vector of these phases by $\mathcal{Y}(\sigma) = (Y_v(\sigma))_{v\in H} \in [q]^H$, we refer to $\mathcal{Y}(\sigma)$ as the phase vector for σ.

For $\mathcal{U} \in [q]^H$, let $\Omega_{\mathcal{U}}$ denote the set of configurations σ on H^G where $\mathcal{Y}(\sigma) = \mathcal{U}$. Let $Z_{H^G}(\mathcal{U})$ be the partition function of H^G restricted to configurations $\sigma \in \Omega_{\mathcal{U}}$, that is,

$$Z_{H^G}(\mathcal{U}) = \sum_{\sigma\in\Omega_{\mathcal{U}}} B^{m(\sigma)},$$

where for a configuration σ, $m(\sigma)$ is the number of monochromatic edges under σ. We may view $\mathcal{U}$ as an assignment $V(H) \to [q]$ where $V(H)$ are the vertices in the graph H. Hence, we can consider the number of monochromatic edges in the graph H under the assignment $\mathcal{U}$, which we denote by $m(\mathcal{U})$. Recall the goal is to analyze $\frac{Z_{H^G}(B)}{Z_{\hat{H}^G}(B)}$. To this end we will analyze $\frac{Z_{H^G}(\mathcal{U})}{Z_{\hat{H}^G}(\mathcal{U})}$ for every $\mathcal{U}$ and then we will use that every $\mathcal{U}$ is (close to) equally likely in $\hat{H}^G$ which will follow from Property 1 in Lemma 8. Notice that once we fix an assignment to all of the roots in R_H then the gadgets G_v are independent of each other. Hence we have that:

$$\frac{Z_{H^G}(\mathcal{U})}{Z_{\hat{H}^G}(\mathcal{U})} = \sum_{\sigma_{R_H}} \mu_{\hat{H}^G}(\sigma_{R_H} \,|\, \mathcal{Y}(\sigma) = \mathcal{U}) \prod_{(u,v)\in E(H^G)\setminus E(\hat{H}^G)} B^{\mathbf{1}\{\sigma_{R_H}(u)=\sigma_{R_H}(v)\}}.$$

Note that $\mu_{\hat{H}^G}(\sigma_{R_H} \,|\, \mathcal{Y}(\sigma) = \mathcal{U}) = \big(1 + O(n^{-\theta})\big) \prod_{v\in V(H)} Q^{\mathcal{U}_v}_{R_v}(\sigma_{R_v})$ since $\hat{H}^G$ is a union of disconnected copies of G and in each copy of G we have Property 2 of Lemma 8. It follows that

$$\begin{aligned}\frac{Z_{H^G}(\mathcal{U})}{Z_{\hat{H}^G}(\mathcal{U})} &= \big(1 + O(n^{-\theta})\big) \sum_{\sigma_{R_H}} \prod_{v\in V(H)} Q^{\mathcal{U}_v}_{R_v}(\sigma_{R_v}) \prod_{(u,v)\in E(H^G)\setminus E(\hat{H}^G)} B^{\mathbf{1}\{\sigma_{R_H}(u)=\sigma_{R_H}(v)\}} \\ &= \big(1 + O(n^{-\theta})\big) A^{m(\mathcal{U})} D^{|E(H)|-m(\mathcal{U})},\end{aligned}$$

where A (resp. D) is the expected weight of an edge for two gadgets which have the same (resp. different) phases. Simple calculations show that

$$A = 1 + (B-1)\Big(p^2 + \frac{(1-p)^2}{q-1}\Big),\ D = 1 + (B-1)\Big(\frac{2p(1-p)}{(q-1)^2} + (q-2)\frac{(1-p)^2}{(q-1)^2}\Big).$$

Letting $B^* = A/D$ and $C_H = D^{|E(H)|}$, we obtain

$$\frac{Z_{H^G}(\mathcal{U})}{Z_{\hat{H}^G}(\mathcal{U})} = \big(1 + O(n^{-\theta})\big)(B^*)^{m(\mathcal{U})} C_H. \tag{8}$$

Property 1 in Lemma 8 gives that for every $\mathcal{U}$ it holds that

$$\big(1 - O(n^{-\theta})\big)q^{-n'} \le \Big(\frac{1}{q} - n^{-2\theta}\Big)^{n'} \le \frac{Z_{\hat{H}^G}(\mathcal{U})}{Z_{\hat{H}^G}} \le \Big(\frac{1}{q} + n^{-2\theta}\Big)^{n'} \le \big(1 + O(n^{-\theta})\big)q^{-n'}. \tag{9}$$

We also have

$$Z_{H^G}(B) = \sum_{\mathcal{U}} Z_{H^G}(\mathcal{U}) = \sum_{\mathcal{U}} \frac{Z_{H^G}(\mathcal{U})}{Z_{\hat{H}^G}(\mathcal{U})} Z_{\hat{H}^G}(\mathcal{U}) = Z_{\hat{H}^G} \sum_{\mathcal{U}} \frac{Z_{H^G}(\mathcal{U})}{Z_{\hat{H}^G}(\mathcal{U})} \frac{Z_{\hat{H}^G}(\mathcal{U})}{Z_{\hat{H}^G}}. \tag{10}$$

Using the estimates (8), (9) in (10), we obtain

$$\big(1 - O(n^{-\theta})\big) q^{-n'} C_H Z_H(B^*) \leq \frac{Z_{H^G}(B)}{Z_{\hat{H}^G}(B)} \leq \big(1 + O(n^{-\theta})\big) q^{-n'} C_H Z_H(B^*).$$

The result follows after observing that $Z_{\hat{H}^G}(B) = \big(Z_G(B)\big)^{n'}$ and rearranging the inequality. ◀

5 Bethe Prediction for Ferromagnetic Models on Random Δ-regular Graphs

5.1 Small Subraph Conditioning Method

By Theorem 5, we have that for the random variable $Z_G^{\boldsymbol{\alpha}}$, when $\boldsymbol{\alpha}$ is a global maximizer of Ψ_1, the exponential order of its second moment is twice the exponential order of its first moment. This is not sufficient however to obtain high probability results, since it turns out that, in the limit $n \to \infty$, the ratio of the second moment to the square of the first moment converges to a constant greater than 1. Hence, the second moment method fails to give statements that hold with high probability over a uniform random Δ-regular graph. More specifically, to obtain our results we need sharp lower bounds on the partition function which hold for almost all Δ-regular graphs. In the setting we described, the second moment method only implies the existence of a graph which satisfies the desired bounds and even there in a not sufficiently strong form.

For random Δ-regular graph ensembles, the standard way to circumvent this failure is to use the small subgraph conditioning method of Robinson and Wormald [34]. While the method is quite technical, its application is relatively streamlined when employed in the right framework. The method was first used for the analysis of spin systems in the work of [32] for the hard-core model and subsequently in [35], [16]. In [17], we extended the approach to q-spin models for all $q \geq 2$, where the major technical obstacle was the computation of certain determinants which arise in the computation of the moments' asymptotics. While the arguments there are for random *bipartite* Δ-regular graphs, the approach extends in a straightforward manner to random Δ-regular graphs.

We defer the details of the application of the method in the present setting to the full version of the paper, see [15, Section 11.1]. We state here the following lemma which is the final outcome of the method.

▶ **Lemma 10.** *For every ferromagnetic model* $\mathbf{B}$*, if* $\boldsymbol{\alpha}$ *is a Hessian dominant phase (c.f. Section 3.2) with probability* $1 - o(1)$ *over the choice of the graph* $G \sim \mathcal{G}(n, \Delta)$*, it holds that* $Z_G^{\boldsymbol{\alpha}} \geq \frac{1}{n} \mathbf{E}\big[Z_G^{\boldsymbol{\alpha}}\big]$.

5.2 Proof of Theorem 6

Using Lemma 10, the proof of Theorem 6 is straightforward.

Proof of Theorem 6. Let $\boldsymbol{\alpha}$ be a Hessian dominant phase, whose existence is guaranteed by the assumptions. By Lemma 10, with probability $1 - o(1)$ over the choice of the graph, we have $Z_G^{\boldsymbol{\alpha}} \geq \frac{1}{n} \mathbf{E}\big[Z_G^{\boldsymbol{\alpha}}\big]$, which implies $\frac{1}{n} \log Z_G \geq \Psi_1(\boldsymbol{\alpha}) + o(1)$.

Moreover, since the model is ferromagnetic, for Δ-regular graphs G with n vertices, $\frac{1}{n}\log Z_G \geq C$ for some constant $C > -\infty$ (explicitly, one can take $C := \frac{\Delta}{2}\log\max_{i\in[q]} B_{ii}$, see the remarks after Theorem 6). We thus obtain

$$\liminf_{n\to\infty} \frac{1}{n}\mathbf{E}_{\mathcal{G}}[\log Z_G] \geq \liminf_{n\to\infty} \big[(1-o(1))\Psi_1(\boldsymbol{\alpha}) + o(1)C\big] = \Psi_1(\boldsymbol{\alpha}).$$

By Jensen's inequality, we also have

$$\limsup_{n\to\infty} \frac{1}{n}\mathbf{E}_{\mathcal{G}}[\log Z_G] \leq \lim_{n\to\infty} \frac{1}{n}\log \mathbf{E}_{\mathcal{G}}[Z_G].$$

All that remains to show is that $\frac{1}{n}\log \mathbf{E}_{\mathcal{G}}[Z_G] = \Psi_1(\boldsymbol{\alpha}) + o(1)$. This is straightforward; if we decompose Z_G as $Z_G = \sum_{\boldsymbol{\alpha}'} Z_G^{\boldsymbol{\alpha}'}$, we obtain $\exp(o(n))\mathbf{E}_{\mathcal{G}}[Z_G^{\boldsymbol{\alpha}}] \geq \mathbf{E}_{\mathcal{G}}[Z_G] \geq \mathbf{E}_{\mathcal{G}}[Z_G^{\boldsymbol{\alpha}}]$. Note the $\exp(o(n))$ is there to allow for dominant phases which are not Hessian.

This concludes the proof. ◀

References

1. G. Bennett. Schur multipliers. *Duke Mathematical Journal*, 44(3):603–639, 1977.
2. A. A. Bulatov, M. Dyer, L. A. Goldberg, M. Jerrum, and C. McQuillan. The expressibility of functions on the boolean domain, with applications to counting CSPs. *Journal of the ACM*, 60(5):Article No. 32, October 2013.
3. C. Borgs, J. T. Chayes, A. Frieze, J. H. Kim, P. Tetali, E. Vigoda, and V. H. Vu. Torpid mixing of some Monte Carlo Markov chain algorithms in statistical physics. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 218–229, 1999.
4. C. Borgs, J. T. 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. G. R. Brightwell and P. Winkler. Random colorings of a Cayley tree. In *Contemporary combinatorics*, volume 10 of *Bolyai Soc. Math. Stud.*, pages 247–276. János Bolyai Math. Soc., Budapest, 2002.
6. J.-Y. Cai, A. Galanis, L. A. Goldberg, H. Guo, M. Jerrum, D. Štefankovič, and E. Vigoda. #BIS-hardness for 2-spin systems on bipartite bounded degree graphs in the tree non-uniqueness region. Preprint available from the arXiv at: http://arxiv.org/abs/1311.4451
7. X. Chen, M. E. Dyer, L. A. Goldberg, M. Jerrum, P. Lu, C. McQuillan, and D. Richerby. The complexity of approximating conservative counting CSPs. In *Proceedings of the Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 148–159, 2013.
8. C. Cooper and A. M. Frieze. Mixing properties of the Swendsen-Wang process on classes of graphs. *Random Structures and Algorithms*, 15(3-4):242–261, 1999.
9. M. Costeniuc, R. S. Ellis, and H. Touchette. Complete analysis of phase transitions and ensemble equivalence for the Curie-Weiss-Potts model. *Journal of Mathematical Physics*, 46(6):paper 063301, 2005.
10. A. Dembo and A. Montanari. Ising models on locally tree-like graphs. *The Annals of Applied Probability*, 20(2):565–592, 2010.
11. A. Dembo, A. Montanari, A. Sly, and N. Sun. The replica symmetric solution for Potts models on d-regular graphs. *Communications in Mathematical Physics* (to appear). Preprint is available from the arXiv at: http://arxiv.org/abs/1207.5500.
12. A. Dembo, A. Montanari, and N. Sun. Factor models on locally tree-like graphs. *The Annals of Applied Probability* (to appear). Preprint is available from the arXiv at: http://arxiv.org/abs/1110.4821.

13 M. E. Dyer, L. A. Goldberg, C. S. Greenhill, and M. Jerrum. The relative complexity of approximate counting problems. *Algorithmica*, 38(3):471–500, 2003.

14 A. Galanis, Q. Ge, D. Štefankovič, E. Vigoda, and L. Yang. Improved inapproximability results for counting independent sets in the hard-core model. *Random Struct. Algorithms* (to appear). Preprint is available from the arXiv at: http://arxiv.org/abs/1105.5131

15 A. Galanis, D. Štefankovič, E. Vigoda, and L. Yang. Ferromagnetic Potts model: Refined #BIS-hardness and related results. Full version of this paper is available from the arXiv at: http://arxiv.org/abs/1311.4839

16 A. Galanis, D. Štefankovič, and E. Vigoda. Inapproximability of the partition function for the antiferromagnetic Ising and hard-core models. Preprint is available from the arXiv at: http://arxiv.org/abs/1203.2226

17 A. Galanis, D. Štefankovič, and E. Vigoda. Inapproximability for antiferromagnetic spin systems in the tree non-uniqueness region. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC)*, 823–831, 2014. Full version is available from the arXiv at: http://arxiv.org/abs/1305.2902

18 H.-O. Georgii. *Gibbs measures and phase transitions*, volume 9 of *de Gruyter Studies in Mathematics*. Walter de Gruyter & Co., Berlin, second edition, 2011.

19 A. Gerschenfeld and A. Montanari. Reconstruction for models on random graphs. In *Proceedings of the 48th Annual Symposium on Foundations of Computer Science* (FOCS), 194-204, 2007.

20 L. A. Goldberg and M. Jerrum. Approximating the partition function of the ferromagnetic Potts model. *Journal of the ACM*, 59(5):Article No. 25, 2012.

21 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.

22 L. A. Goldberg, M. Jerrum, and M. Paterson. The computational complexity of two-state spin systems. *Random Struct. Algorithms*, 23(2):133–154, 2003.

23 C. Greenhill. The complexity of counting colourings and independent sets in sparse graphs and hypergraphs. *Comput. Complex.*, 9(1):52–72, 2000.

24 O. Häggström. The random-cluster model on a homogeneous tree. *Probability Theory and Related Fields*, 104(2):231–253, 1996.

25 S. Janson. Random regular graphs: Asymptotic distributions and contiguity. *Combinatorics, Probability & Computing*, 4:369–405, 1995.

26 M. Jerrum and A. Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal on Computing*, 22(5):1087–1116, 1993.

27 F. P. Kelly. Loss networks. *The Annals of Applied Probability*, 1(3):319–378, 1991.

28 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 47–66. 2013.

29 F. Martinelli, A. Sinclair, and D. Weitz. Fast mixing for independent sets, colorings and other models on trees. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 456–465. 2004.

30 R. Montenegro and P. Tetali. Mathematical aspects of mixing times in Markov chains. *Foundations and Trends in Theoretical Computer Science*, 1(3):237–354, 2006.

31 E. Mossel and A. Sly. Exact thresholds for Ising-Gibbs samplers on general graphs. *Annals of Probability*, 41(1):294–328, 2013.

32 E. Mossel, D. Weitz, and N. Wormald. On the hardness of sampling independent sets beyond the tree threshold. *Probability Theory and Related Fields*, 143(3-4):401–439, 2009.

33 F. Peruggi, F. Di Liberto, and G. Monroy. Phase diagrams of the q-state Potts model on Bethe lattices. *Physica A*, 141(1):151–186, 1987.

34 R. W. Robinson and N. C. Wormald. Almost all regular graphs are Hamiltonian. *Random Structures and Algorithms*, 5(2):363–374, 1994.

35 A. Sly. Computational transition at the uniqueness threshold. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 287–296, 2010.

36 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.

37 M. Ullrich. Rapid mixing of Swendsen-Wang dynamics in two dimensions. Ph.D. Thesis, Universität Jena, Germany, 2012. The thesis is available from the arXiv at: http://arxiv.org/abs/1212.4908

38 L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Computing*, 8(3):410–421, 1979.

39 D. Weitz. Counting independent sets up to the tree threshold. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, 140–149, 2006.

Space Pseudorandom Generators by Communication Complexity Lower Bounds*

Anat Ganor[1] and Ran Raz[1,2]

1 **Weizmann Institute of Science, Rehovot, Israel**
{anat.ganor,ran.raz@weizmann.ac.il}@weizmann.ac.il
2 **Institute for Advanced Study, Princeton, New Jersey**

Abstract

In 1989, Babai, Nisan and Szegedy [2] gave a construction of a pseudorandom generator for logspace, based on lower bounds for multiparty communication complexity. The seed length of their pseudorandom generator was $2^{\Theta(\sqrt{\log n})}$, because the best lower bounds for multiparty communication complexity are relatively weak. Subsequently, pseudorandom generators for logspace with seed length $O(\log^2 n)$ were given by [19] and [15].

In this paper, we show how to use the pseudorandom generator construction of [2] to obtain a third construction of a pseudorandom generator with seed length $O(\log^2 n)$, achieving the same parameters as [19] and [15]. We achieve this by concentrating on protocols in a restricted model of multiparty communication complexity that we call the *conservative one-way unicast model* and is based on the conservative one-way model of [8]. We observe that bounds in the conservative one-way unicast model (rather than the standard Number On the Forehead model) are sufficient for the pseudorandom generator construction of [2] to work.

Roughly speaking, in a conservative one-way unicast communication protocol, the players speak in turns, one after the other in a fixed order, and every message is visible only to the next player. Moreover, before the beginning of the protocol, each player only knows the inputs of the players that speak after she does and a certain function of the inputs of the players that speak before she does. We prove a lower bound for the communication complexity of conservative one-way unicast communication protocols that compute a family of functions obtained by compositions of strong extractors. Our final pseudorandom generator construction is related to, but different from the constructions of [19] and [15].

1998 ACM Subject Classification F.1.0 Computation by Abstract Devices – General

Keywords and phrases Communication complexity, Logspace, Pseudorandom generator

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.692

1 Introduction

Derandomizing space bounded computations has attracted a lot of attention over the last few decades. The most important problem is to simulate randomized logspace machines with deterministic ones. Savitch [26] result on nondeterministic machines implies that $RL \subseteq L^2$. Subsequently, this problem was studied, for example, by [1], [2], [19], [15] and [22]. Currently, the best derandomization of general logspace machines is due to Saks and Zhou [25], proving that $BPL \subseteq L^{3/2}$.

* This work was supported by an ISF grant, by the I-CORE Program of the Planning and Budgeting Committee and by NSF grant numbers CCF-0832797, DMS-0835373.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 692–703

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

One way to simulate a randomized, space bounded computation with a deterministic one is using a space pseudorandom generator. Roughly speaking, a space pseudorandom generator converts, efficiently, a short truly random seed into a long string that looks random to machines with limited space. A major open problem in the theory of pseudorandomness is to construct an explicit pseudorandom generator that stretches a seed of length $O(\log n)$ to n bits that cannot be distinguished from uniform by any logspace machine with input length n. Such a generator would imply that $RL = L$. Nisan [19] constructed space pseudorandom generators that convert $O(\log^2 n)$ random bits to $poly(n)$ bits that look random to any logspace machine. Subsequently, [15] showed a different construction with the same parameters. Since [19] and [15], no better seed length was obtained for derandomizing general logspace machines. There were other constructions of space pseudorandom generators for more restricted classes of space bounded computations, such as [23], [5], [6], [18], [14], [3] and [24].

In this paper, we give a new construction of a space pseudorandom generator for general logspace machines, with seed length $O(\log^2 n)$, achieving the same parameters as [19] and [15]. Our pseudorandom generator construction is based on a lower bound for a certain model of multiparty communication complexity, relying on the pseudorandom generator construction of Babai, Nisan and Szegedy [2]. The pseudorandom generator of [2] has seed length $2^{\Theta(\sqrt{\log n})}$. The proof that their construction gives a pseudorandom generator relies on a lower bound for multiparty communication complexity. [2] gave a lower bound for the multiparty communication complexity of protocols in the Number On the Forehead (NOF) model with blackboard communication. In this model, each player knows all inputs except her own input and the communication is done by writing messages on a blackboard (broadcast) so that every player sees all the previous communication. For this model, [2] gave a lower bound of $\Omega(\frac{n}{2^k})$ (where n is the length of each input and k is the number of players). Improving this lower bound is a major open problem.

We observe that the pseudorandom generator construction of [2] can be based on lower bounds for a restricted model of multiparty communication complexity. For this model we are able to obtain improved lower bounds, resulting in a pseudorandom generator with seed length $O(\log^2 n)$.

▶ **Definition 1** (Conservative One-way Unicast Communication Protocol)**.** Let P be a deterministic, multiparty communication protocol for k players $p_1, \ldots, p_k$. For a function $f : \mathcal{B} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_k \to \mathcal{B}$, we say that P is a *conservative one-way unicast communication protocol with respect to* f if for an input $b, a_1, \ldots, a_k \in \mathcal{B} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_k$ the following holds:

1. For every $i \in [k]$, before the beginning of the protocol, the i^{th} player only knows $a_{i+1}, \ldots, a_k$ and the (truth table[1] of the) function $f_i : \mathcal{A}_i \times \cdots \times \mathcal{A}_k \to \mathcal{B}$, defined by:

$$f_i(z_i, \ldots, z_k) = f(b, a_1, \ldots, a_{i-1}, z_i, \ldots, z_k)$$

   for every $z_i, \ldots, z_k \in \mathcal{A}_i \times \cdots \times \mathcal{A}_k$.
2. The players communicate one after the other in the fixed order $p_1, p_2, \ldots, p_k$.
3. For every $1 \leq i < k$, the i^{th} message is visible only to p_{i+1}. The message of the last player is the output of the protocol.

Usually, we will take f to be the function that the players are trying to compute. Note that the i^{th} player doesn't know $b, a_1, \ldots, a_{i-1}$ as in the NOF model, but she does know the relevant in-

[1] The truth table is not counted as part of the length of the input.

formation on $b, a_1, \ldots, a_{i-1}$ that is needed to compute the function $f(b, a_1, \ldots, a_{i-1}, z_i, \ldots, z_k)$ for every $z_i, \ldots, z_k \in \mathcal{A}_i \times \cdots \times \mathcal{A}_k$.

Our definition of conservative one-way unicast communication protocols is based on definitions by Damm, Jukna and Sgall [8]. [8] defined conservative communication protocols as protocols satisfying item (1) in Definition 1, and conservative one-way communication protocols as protocols satisfying items (1),(2) in Definition 1, where the communication is done by writing messages on a blackboard (broadcast) so that every player sees all the previous communication. The motivation of [8] to study the conservative one-way model was different than ours. They studied this model as an interesting communication model in its own right, without relating it to pseudorandom generators for logspace computations.

[8] proved lower bounds for the communication complexity of conservative one-way (blackboard) communication protocols that compute the pointer jumping problem. For $k = O((n/\log n)^{1/3})$, [8] proved a lower bound of $\Omega(n/k^2)$, and for $k \leq \log^* n - \omega(1)$, they proved a lower bound of $n \log^{(k-1)} n(1 - o(1))$ (where k is the number of players and n is the length of each input). The conservative one-way model was further studied by Chakrabarti in [7], where the $\Omega(n/k^2)$ lower bound due to [8] was extended so that it applies for all k.

The unicast setting, where the players communicate by sending messages to each other over private channels, was studied in the context of message-passing models of multiparty communication. These models have been used extensively in distributed computing, for example in [12], [16], [17], and [4]. Message passing models are also used to study privacy and security in multiparty computations.

For conservative communication protocols (satisfying item (1) in Definition 1) it is convenient to consider composed functions as we define next.

▶ **Definition 2** (Composed Functions)**.** For a function $f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^m$ and $1 < i \in \mathbb{N}$, the i^{th} *composition of* f is a function $f^{(i)} : \{0,1\}^{m+in} \to \{0,1\}^m$ defined for every $a_0 \in \{0,1\}^m$, $a_1, \ldots, a_i \in \{0,1\}^n$ as

$$f^{(i)}(a_0, a_1, \ldots, a_i) = f(f^{(i-1)}(a_0, a_1, \ldots, a_{i-1}), a_i)$$

where $f^{(1)}(a_0, a_1) = f(a_0, a_1)$. In addition, we define $f^{(0)}(a_0) = a_0$.

Let $f^{(k)}$ be the k^{th} composition of a function $f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^m$. Note that for every input $(a_0, a_1, \ldots, a_k) \in \{0,1\}^{m+kn}$ and every $i \in [k]$, if the i^{th} player knows $f^{(i-1)}(a_0, a_1, \ldots, a_{i-1})$, then she also knows the function $f_i^{(k)}$ defined in item (1) in Definition 1. Therefore, for the sake of proving lower bounds for the communication complexity of conservative communication protocols with respect to $f^{(k)}$, it is enough to assume that for an input $(a_0, a_1, \ldots, a_k) \in \{0,1\}^{m+kn}$, for every $i \in [k]$, before the beginning of the protocol, the i^{th} player only knows $a_{i+1}, \ldots, a_k$ and $f^{(i-1)}(a_0, a_1, \ldots, a_{i-1})$.

In our paper, we prove lower bounds for the communication complexity of conservative one-way unicast communication protocols with respect to a certain composed function $f^{(k)}$. Therefore, we replace item (1) in Definition 1 by the assumption that for an input $(a_0, a_1, \ldots, a_k) \in \{0,1\}^{m+kn}$, for every $i \in [k]$, before the beginning of the protocol, the i^{th} player only knows $a_{i+1}, \ldots, a_k$ and $f^{(i-1)}(a_0, a_1, \ldots, a_{i-1})$.

An Example – The Pointer Jumping Problem

In the pointer jumping problem for k players, the input is k functions $\Pi_1, \ldots, \Pi_k : [r] \to [r]$ and an additional input $i_0 \in [r]$. The players need to output $\Pi_k \circ \cdots \circ \Pi_1(i_0)$. Let $\mathcal{S}_r$ denote the set of all functions from $[r]$ to $[r]$ and let $f : [r] \times \mathcal{S}_r \to [r]$ be the function defined by

$f(i, \Pi) = \Pi(i)$ for every $i \in [r]$ and $\Pi \in S_r$. Note that the pointer jumping problem for k players is the k^{th} composition of f. In a conservative communication protocol (satisfying item (1) in Definition 1) for the pointer jumping problem, for every $i \in [k]$, before the beginning of the protocol, the i^{th} player only knows $\Pi_{i+1}, \ldots, \Pi_k$ and $f^{i-1}(i_0, \Pi_1, \ldots, \Pi_{i-1}) = \Pi_{i-1} \circ \cdots \circ \Pi_1(i_0)$.[2]

1.1 Main Result

We say that a communication protocol P computes a function $f : \{0,1\}^n \to \{0,1\}^m$ with bias $\delta > 0$ if

$$\Pr_{x \in_R \{0,1\}^n}[f(x) = P(x)] \geq 2^{-m} + \delta$$

We denote the length of the longest message sent during the execution of P by $L(P)$ (on the worst case input, not including the last message which is the output of the protocol).

Let $Ext : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^m$ be a (t, ε) strong extractor (see Definition 10). We refer to the k^{th} composition of Ext, denoted $Ext^{(k)}$, as a (t, ε) *composed strong extractor*. Composed strong extractors are closely related to alternating extractors, which are used in [10], with cryptographic applications.

Our lower bound is for the length of the longest message communicated during any conservative one-way unicast communication protocol that computes a composed strong extractor with bias $\delta > 0$.

▶ **Theorem 3.** *Let $Ext^{(k)} : \{0,1\}^{m+nk} \to \{0,1\}^m$ be a (t, ε) composed strong extractor and let P be a conservative one-way unicast communication protocol with respect to $Ext^{(k)}$ that computes $Ext^{(k)}$ with bias $\delta > 0$, such that $\varepsilon < \delta \cdot 2^{-(k+2)}$. Then,*

$$L(P) \geq n - t - k - \log \frac{1}{\delta} - 2$$

In fact, we prove a slightly stronger version of Theorem 3 in which we consider projections of the composed strong extractor (see Theorem 17). Using this lower bound together with the pseudorandom generator construction of Babai, Nisan and Szegedy [2], we obtain a space pseudorandom generator that converts $O(\log^2 n)$ random bits to $poly(n)$ bits that look random to any logspace machine (see Section 4).

Comparison with [19] and [15]

The pseudorandom generator construction of [15] is also based on a recursive composition of extractors. However, their generator is different from the one presented here. The recursive composition used in [15] is different from the composition in Definition 2. Moreover, [15] use extractors that output $O(\log^2 n)$ bits, whereas here we use extractors that output $O(\log n)$ bits.

The pseudorandom generator construction of [19] is based on a recursive composition of hash functions. This is done by a composition similar to the one in Definition 2. We note that hashing can be viewed as an application of an extractor. However, when viewing the hashing as an application of an extractor, the composition of [19] does not fit our definition of a composed extractor. In particular, in our definition of a composed extractor, the recursion

[2] In this case, knowing $\Pi_{i-1} \circ \cdots \circ \Pi_1(i_0)$ is equivalent to knowing the function f_i defined in item (1) in Definition 1.

is done by replacing the seed of the extractor with the output of the extractor from the previous composition, whereas in [19], the recursion is done by replacing the source of the extractor with the output of the extractor from the previous composition.

2 Preliminaries

2.1 General Notation

Let $[n]$ be the set of numbers $\{1, 2, \ldots, n\}$. For a binary string $x \in \{0,1\}^*$ and an index $i \in \mathbb{N}$, let x_i be the i^{th} bit of x. For a set of indexes $S = \{i_1, \ldots, i_k\} \subseteq [|x|]$, let x_S be the string $x_{i_1}, \ldots, x_{i_k}$.

2.1.1 Functions

For a function $f : \{0,1\}^n \to \{0,1\}^m$ and a subset $S \subseteq [m]$ of size m', where $m' \leq m$, the projection of f on S, denoted f_S, is a function from $\{0,1\}^n$ to $\{0,1\}^{m'}$ defined as $f_S(x) = (f(x))_S$ for every $x \in \{0,1\}^n$. To simplify notation, for $i \in [m]$, we define $f_i = f_{\{i\}}$. For two functions $f : \mathcal{A} \to \mathcal{B}$ and $h : \mathcal{B} \to \mathcal{C}$, let $h \circ f$ be the function from $\mathcal{A}$ to $\mathcal{C}$ defined as $h(f(a))$ for every $a \in \mathcal{A}$.

2.1.2 Distributions and Random Variables

We write $x \in_R \mathcal{X}$ if x is chosen uniformly at random from $\mathcal{X}$. For a distribution D and a subset S of the support of D, let $D(S)$ be the sum $\sum_{s \in S} D(s)$. For a random variable X and an event E, we write $X|E$ to denote X conditioned on E. We write $X \in \mathcal{X}$ if X is distributed over the set $\mathcal{X}$. For two random variables X and Y, we write $X \sim Y$ if X and Y have the same distribution. Slightly abusing notation, given a random variable X, we let $x \sim X$ indicate the sampling of x from the distribution of X.

2.2 Statistical Distance

▶ **Definition 4** (Statistical Distance)**.** Let D_1 and D_2 be two distributions over the same space Ω. Their *statistical distance* is

$$\|D_1 - D_2\| = \max_{S \subseteq \Omega} |D_1(S) - D_2(S)| = \frac{1}{2} \sum_{x \in \Omega} |D_1(x) - D_2(x)|$$

For two random variables $X_1, X_2 \in \Omega$ distributed according to D_1 and D_2 respectively, we define $\|X_1 - X_2\| = \|D_1 - D_2\|$.

▶ **Proposition 5.** *Let $X, X' \in \mathcal{X}$ be two random variables and let $f : \mathcal{X} \to \mathcal{Y}$ be any deterministic function. Then,*

$$\|f(X) - f(X')\| \leq \|X - X'\|$$

▶ **Proposition 6.** *Let $X \in \mathcal{X}$, $Y \in \mathcal{Y}$ and $Z \in \mathcal{Z}$ be three random variables, and let U be uniform over $\mathcal{X}$, independent of X, Y and Z. Then,*

$$\|(Z, X) - (Z, U)\| \leq \|(Y, Z, X) - (Y, Z, U)\|$$

2.3 Space Pseudorandom Generators

A deterministic, space $s(n)$ Turing machine uses $s(n)$ space on any input of size n. A non-uniform, space $s(n)$ statistical test is a deterministic, space $s(n)$ Turing machine M and an infinite sequence of binary strings $a = (a_1, \ldots, a_n, \ldots)$ called the advice strings, where the length of a_n is $\exp(s(n))$, for every $n \in \mathbb{N}$. The result of the test on input x, denoted $M^a(x)$, is the result of running M on x when it has access to the advice $a_{|x|}$. The machine M reads the advice as if it is on a normal input tape, and it has a one-way access to the input x (i.e., it can access the next bit of x but it cannot go "back" and review bits it already read). A pseudorandom generator for space bounded computations is required to produce strings that can be used instead of truly random strings in randomized, space bounded computations (while introducing only small additional error). Therefore, a pseudorandom generator must produce strings that look random to any non-uniform, bounded space statistical test. The following is a formal definition. For more information see e.g. [2].

▶ **Definition 7.** $G = \left\{G_n : \{0,1\}^{m(n)} \to \{0,1\}^n\right\}$ is an ε pseudorandom generator for space $s(n)$ if for every non-uniform, space $s(n)$ statistical test M^a it holds that

$$\left| \Pr_{x \in_R \{0,1\}^n}[M^a(x) = 1] - \Pr_{y \in_R \{0,1\}^{m(n)}}[M^a(G(y)) = 1] \right| \leq \varepsilon$$

The following is an alternative definition (which is equivalent upto a multiplicative factor of n change in ε).

▶ **Definition 8.** $G = \left\{G_n : \{0,1\}^{m(n)} \to \{0,1\}^n\right\}$ is an ε pseudorandom generator for space $s(n)$ if for every $i \in [n]$ and for every non-uniform, space $s(n)$ statistical test M^a it holds that

$$\left| \Pr_{y \in_R \{0,1\}^{m(n)}}[M^a(\text{first } i-1 \text{ bits of } G(y)) = i^{th} \text{ bit of } G(y)] - \frac{1}{2} \right| \leq \varepsilon$$

In this paper, we use Definition 8.

2.4 Strong Extractors

The notion of weak source was first defined by Nisan and Zuckerman [21].

▶ **Definition 9** (Min-Entropy)**.** For a random variable X, the *min-entropy* of X is

$$\mathbf{H}_\infty(X) = -\log \max_x \Pr[X = x]$$

An (n,t) source is a random variable in $\{0,1\}^n$ that has min-entropy at least t.

▶ **Definition 10** (Strong Extractor [22])**.** A function $Ext : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^\ell$ is a (t,ε) strong extractor if for every (n,t) source X and every seed S uniformly distributed over $\{0,1\}^m$ it holds that

$$\|(S, Ext(S,X)) - U\| \leq \varepsilon$$

where U is uniformly distributed over $\{0,1\}^{m+\ell}$.

2.4.1 Average Min-Entropy and Average-Case Extractors

The following definitions and lemmas appear in [9].

▶ **Definition 11** (Average Min-Entropy). For two random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$, the *average min-entropy* of X given Y is

$$\tilde{\mathbf{H}}_\infty(X|Y) = -\log \mathop{\mathrm{E}}_{y\sim Y} \max_{x\in\mathcal{X}} \Pr[X = x|Y = y] = -\log \mathop{\mathrm{E}}_{y\sim Y} \left[2^{-\mathbf{H}_\infty(X|Y=y)}\right]$$

▶ **Lemma 12.** *Let X, Y and Z be random variables. If Y has at most 2^ℓ possible values, then*

$$\tilde{\mathbf{H}}_\infty(X|(Y,Z)) \geq \tilde{\mathbf{H}}_\infty((X,Y)|Z) - \ell \geq \tilde{\mathbf{H}}_\infty(X|Z) - \ell$$

▶ **Definition 13** (Average-case Strong Extractor). A function $Ext : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^\ell$ is an average-case (t,ε) strong extractor if for every pair of random variables (W, I) such that $W \in \{0,1\}^n$ and $\tilde{\mathbf{H}}_\infty(W|I) \geq t$, and every seed S uniformly distributed over $\{0,1\}^m$, it holds that

$$\|(I, S, Ext(S, W)) - (I, U)\| \leq \varepsilon$$

where U is uniformly distributed over $\{0,1\}^{m+\ell}$.

▶ **Lemma 14.** *For any $\gamma > 0$, if Ext is a $(t - \log 1/\gamma, \varepsilon)$ strong extractor, then Ext is also an average-case $(t, \varepsilon + \gamma)$ strong extractor.*

3 Lower Bounds for Conservative One-way Unicast Communication Protocols

For the construction of the pseudorandom generator in Section 4, we will need a lower bound for the communication complexity of a function that outputs a single bit. To this end we consider also projections of composed strong extractors (see notation for projections in Section 2.1).

▶ **Definition 15.** A function $g : \{0,1\}^{m+nk} \to \{0,1\}^{m'}$ is called a (t,ε) *projection of a composed strong extractor (PCSE)* if $g = Ext^{(k)}_S$, where $S \subseteq [m]$ is a subset of size m' for $m' \leq m$ and $Ext^{(k)} : \{0,1\}^{m+nk} \to \{0,1\}^m$ is a (t,ε) composed strong extractor.

The following lemma is the main technical part of our paper. The proof is related to the proof of the "alternating extraction theorem" in [10], which uses ideas from [11]. See also lecture notes [29].

▶ **Lemma 16.** *Let $Ext^{(k)}_S : \{0,1\}^{m+nk} \to \{0,1\}^{m'}$ be a (t,ε) PCSE and let $A_0 \in \{0,1\}^m$, $A_1, \ldots, A_k \in \{0,1\}^n$ be uniformly and independently distributed. Let P be a conservative one-way unicast communication protocol with respect to $Ext^{(k)}$, and let $M_1, \ldots, M_k$ be the messages sent during the execution of P on inputs $A_0, \ldots, A_k$, where the i^{th} player sends M_i, for $i \in [k]$. Fix $\gamma > 0$ and assume that $M_1, \ldots, M_{k-1} \in \{0,1\}^\ell$, for $\ell \leq n - t - \log \frac{1}{\gamma}$. Then,*

$$\|(M_k, Ext^{(k)}_S(A_0, A_1, \ldots, A_k)) - (M_k, U')\| \leq 2^{k+1}(\varepsilon + \gamma)$$

where U' is uniformly distributed over $\{0,1\}^{m'}$.

Proof. To simplify notation, for every $i \in [k]$ we write $Ext^{(i)}$ instead of $Ext^{(i)}(A_0, A_1, \ldots, A_i)$ and let $\overline{A}_i = (A_i, \ldots, A_k)$. For $i > k$ we define $\overline{A}_i$ to be the empty string. Let U be uniformly distributed over $\{0,1\}^m$. Recall that U' is uniformly distributed over $\{0,1\}^{m'}$. Then, for every $z \in \{0,1\}^{m'}$, by Proposition 5,

$$\|(Ext_S^{(k)}|M_k = z) - U'\| \leq \|(Ext^{(k)}|M_k = z) - U\|$$

Therefore, it is enough to prove that

$$\|(M_k, Ext^{(k)}) - (M_k, U)\| \leq 2^{k+1}(\varepsilon + \gamma) \tag{1}$$

By the definition of a conservative one-way unicast communication protocol with respect to $Ext^{(k)}$, for every $i \in [k]$ it holds that

$$M_i = g_i(\overline{A}_{i+1}, M_{i-1}, Ext^{(i-1)}) \tag{2}$$

where g_i is some (deterministic) function, and $M_0 = 0^\ell$. We prove by induction on i, that for every $0 \leq i \leq k$,

$$\|(\overline{A}_{i+1}, M_i, Ext^{(i)}) - (\overline{A}_{i+1}, M_i, U)\| \leq \sum_{j=0}^{i} 2^j(\varepsilon + \gamma)$$

Substituting $i = k$ we get equation (1) as required. For $i = 0$ we have that $\|(\overline{A}_1, M_0, Ext^{(0)}) - (\overline{A}_1, M_0, U)\| = 0$, and the claim holds. Assume that the claim holds for some $0 \leq i < k$ and let $\Delta = \|(\overline{A}_{i+2}, M_{i+1}, Ext^{(i+1)}) - (\overline{A}_{i+2}, M_{i+1}, U)\|$. By equation (2) and Proposition 5,

$$\Delta \leq \|(\overline{A}_{i+2}, M_i, Ext^{(i)}, Ext^{(i+1)}) - (\overline{A}_{i+2}, M_i, Ext^{(i)}, U)\|$$

By the definition of $Ext^{(i+1)}$,

$$\Delta \leq \|(\overline{A}_{i+2}, M_i, Ext^{(i)}, Ext(Ext^{(i)}, A_{i+1})) - (\overline{A}_{i+2}, M_i, Ext^{(i)}, U)\|$$

Let S be uniformly distributed over $\{0,1\}^m$. By the triangle inequality,

$$\begin{aligned}
&\|(\overline{A}_{i+2}, M_i, Ext^{(i)}, Ext(Ext^{(i)}, A_{i+1})) - (\overline{A}_{i+2}, M_i, Ext^{(i)}, U)\| \leq \\
&\quad \|(\overline{A}_{i+2}, M_i, Ext^{(i)}, Ext(Ext^{(i)}, A_{i+1})) - (\overline{A}_{i+2}, M_i, S, Ext(S, A_{i+1}))\| + && (3)\\
&\quad \|(\overline{A}_{i+2}, M_i, S, Ext(S, A_{i+1})) - (\overline{A}_{i+2}, M_i, S, U)\| + && (4)\\
&\quad \|(\overline{A}_{i+2}, M_i, S, U) - (\overline{A}_{i+2}, M_i, Ext^{(i)}, U)\| && (5)
\end{aligned}$$

By Lemma 14, Ext is also an average-case $(t + \log 1/\gamma, \varepsilon + \gamma)$ strong extractor. By Lemma 12,

$$\tilde{\mathbf{H}}_\infty(A_{i+1}|\overline{A}_{i+2}, M_i) \geq \tilde{\mathbf{H}}_\infty(A_{i+1}|\overline{A}_{i+2}) - \ell = \mathbf{H}_\infty(A_{i+1}) - \ell = n - \ell \geq t + \log 1/\gamma$$

and therefore, by Definition 13, $(4) \leq \varepsilon + \gamma$. By Propositions 5 and 6,

$$(3), (5) \leq \|(\overline{A}_{i+1}, M_i, Ext^{(i)}) - (\overline{A}_{i+1}, M_i, S)\|$$

By the inductive hypothesis, $\|(\overline{A}_{i+1}, M_i, Ext^{(i)}) - (\overline{A}_{i+1}, M_i, S)\| \leq \sum_{j=0}^{i} 2^j(\varepsilon + \gamma)$. Putting it together we get that

$$\Delta \leq \varepsilon + \gamma + 2 \cdot \sum_{j=0}^{i} 2^j(\varepsilon + \gamma) = \sum_{j=0}^{i+1} 2^j(\varepsilon + \gamma)$$

as required. ◀

Finally, we give a lower bound for the length of the longest message in a conservative one-way unicast communication protocol that computes a projection of a composed strong extractor.

▶ **Theorem 17.** *Let* $Ext_S^{(k)} : \{0,1\}^{m+nk} \to \{0,1\}^{m'}$ *be a* (t,ε) *PCSE and let* P *be a conservative one-way unicast communication protocol with respect to* $Ext^{(k)}$ *that computes* $Ext_S^{(k)}$ *with bias* $\delta > 0$*, such that* $\varepsilon < \delta \cdot 2^{-(k+2)}$*. Then,*

$$L(P) \geq n - t - k - \log \frac{1}{\delta} - 2$$

Proof. Let $A_0 \in \{0,1\}^m$, $A_1, \ldots, A_k \in \{0,1\}^n$ be uniformly and independently distributed and let $M_1, \ldots, M_k$ be the messages sent during the execution of the protocol P on inputs $A_0, \ldots, A_k$, where the i^{th} player sends M_i, for $i \in [k]$. To simplify notation, we write $Ext_S^{(k)}$ instead of $Ext_S^{(k)}(A_0, A_1, \ldots, A_k)$. Since the protocol P computes $Ext_S^{(k)}$ with bias δ,

$$\delta + 2^{-m'} \leq \Pr_{\bar{A} \in_R \{0,1\}^{m+nk}} \left[M_k = Ext_S^{(k)} \right]$$

Let U' be uniformly distributed over $\{0,1\}^{m'}$. Since $M_k = Ext_S^{(k)}$ is a statistical test on the distribution $(M_k, Ext_S^{(k)})$, and the same statistical test on (M_k, U') passes with probability $2^{-m'}$,[3]

$$\left| \Pr_{\bar{A} \in_R \{0,1\}^{m+nk}} \left[M_k = Ext_S^{(k)} \right] - 2^{-m'} \right| \leq \|(M_k, Ext_S^{(k)}) - (M_k, U')\|$$

Assume for simplicity and without loss of generality, that all messages $M_1, \ldots, M_{k-1}$ have the same length, denoted ℓ. Fix $\gamma = \delta \cdot 2^{-(k+2)}$ and assume towards a contradiction that $\ell < n - t - \log \frac{1}{\gamma}$. Then, by Lemma 16,

$$\|(M_k, Ext_S^{(k)}) - (M_k, U')\| \leq 2^{k+1}(\varepsilon + \gamma)$$

We get that $\delta \leq 2^{k+1}(\varepsilon + \gamma)$ and therefore, $\gamma \geq \delta \cdot 2^{-(k+1)} - \varepsilon > \delta \cdot 2^{-(k+2)}$, which contradicts our choice of γ. ◀

4 Logspace Pseudorandom Generators

We review the construction of the pseudorandom generator of Babai, Nisan and Szegedy [2]. The generator is based on a function f that takes k arguments, each r bits long, and has high multiparty communication complexity. The ε multiparty communication complexity of f, denoted $C_\varepsilon(f)$, is the communication complexity of the best deterministic communication protocol in the NOF model with blackboard communication that computes f with bias at least ε.

The input to the generator consists of t random strings of length r each. Fix $k \leq t$ and let $S_1, S_2, \ldots, S_{\binom{t}{k}}$ be all k-subsets of the input strings in anti-lexicographic order (i.e., each S_i is a set of k strings, each string is r bits long, and S_i appears before S_j if the last string in the symmetric difference of S_i and S_j belongs to S_j). The output of the generator is $f(S_1), f(S_2), \ldots, f(S_{\binom{t}{k}})$.

The proof of the following lemma appears in [2]. We give it for completeness in Appendix A.

[3] We can assume, without loss of generality, that M_k is of length m'.

▶ **Lemma 18.** *For every $\varepsilon > 0$, every function $f : \{0,1\}^{rk} \to \{0,1\}$ and every $s < C_\varepsilon(f)/k$, the above construction gives an ε pseudorandom generator for space s (see Definition 8).*

We make few observations on Lemma 18 and its proof, that will allow us to use our lower bound from Section 3:

1. In the multiparty communication protocol used for the proof, the players communicate in a fixed order. Hence, we can consider communication protocols that satisfy item (2) in Definition 1.
2. In the multiparty communication protocol used for the proof, for every $i < k$, the i^{th} message, sent by the i^{th} player, is used only by player $i+1$. Therefore, the blackboard is not required and we can consider communication protocols that satisfy item (3) in Definition 1.
3. In the multiparty communication protocol used for the proof, for every $j \in [k]$, if the j^{th} player needs to compute $f(T)$ during the simulation, then it holds that the set T comes before the set S in the anti-lexicographic order, and $y_{i_1}, \ldots, y_{i_{j-1}} \in T$ and $y_{i_j} \notin T$. For every such a set T, the j^{th} player can compute $f(T)$ without knowing $y_{i_1}, \ldots, y_{i_{j-1}}$. It suffices that she knows the strings that were fixed, the input strings $y_{i_{j+1}}, \ldots, y_{i_k}$ and the function $f(y_{i_1}, \ldots, y_{i_{j-1}}, z_j, \ldots, z_k)$ for every $z_j, \ldots, z_k \in \{0,1\}^r$. Hence, we can consider communication protocols that satisfy item (1) in Definition 1.
4. In the multiparty communication protocol used for the proof, all messages have the same length. Hence, we can use a lower bound for the length of the longest message sent during the execution of the protocol.

Note that the function f from Definition 1 has an additional input string $b \in \mathcal{B}$. We can think of b as if it is added to all subsets $S_1, S_2, \ldots, S_{\binom{t}{k}}$. Formally, our **adjusted construction** is as follows. The input to the generator consists of t random strings of length r each and an additional random string $b \in \{0,1\}^m$. Let $S_1, S_2, \ldots, S_{\binom{t}{k}}$ be all k-subsets of the input strings (not including the string b) in anti-lexicographic order, as in the original construction. For every $1 \le j \le \binom{t}{k}$, let $S_j = \{y_{i_{j,1}}, \ldots, y_{i_{j,k}}\}$, where $i_{j,1} > i_{j,2} > \cdots > i_{j,k}$. Then, the j^{th} bit in the output of the generator is $f_1(b, y_{i_{j,1}}, \ldots, y_{i_{j,k}})$. Recall that f_1 returns the first bit of the function f (see notation for projections in Section 2.1).

We get the following lemma.

▶ **Lemma 19.** *Fix $\varepsilon > 0$ and a function $f : \{0,1\}^{m+kr} \to \{0,1\}^m$, such that for every conservative one-way unicast communication protocol with respect to f that computes f_1 with bias ε, the length of the longest message is at least C. Then, for every $s < C$, the adjusted construction gives an ε pseudorandom generator for space s.*

▶ **Corollary 20.** *For every constant $c > 0$, there exists an (explicitly given) n^{-c} pseudorandom generator for logspace which converts $O(\log^2 n)$ random bits to poly(n) bits.*

Proof. Let $m = O(\log n)$, $r = O(\log n)$ and let $f : \{0,1\}^{m+r} \to \{0,1\}^m$ be a (t', ε) strong extractor, such that $t' < r - 2\log n - c\log n - 2$ and $\varepsilon < 1/4n^{c+1}$. For an explicit construction of a strong extractor with such parameters see Theorem 4.2 in [13] (for more information see e.g. [20], [27] and [28]). Let $\delta = n^{-c}$, $k = \log n$ and let P be a conservative one-way unicast communication protocol with respect to $f^{(k)}$ that computes $f_1^{(k)}$ with bias δ. Since $\varepsilon < \delta \cdot 2^{-(k+2)} = 1/4n^{c+1}$, Theorem 17 guarantees that $L(P) \ge r - t' - k - \log\frac{1}{\delta} - 2 = r - t' - \log n - c\log n - 2 > \log n$. By Lemma 19, using the adjusted construction with the PCSE $f_1^{(k)}$ and $t = k \cdot 2^{c'}$ for any constant $c' > 1$, we get a δ pseudorandom generator for space $\log n$, that on a seed of length $m + tr = O(\log^2 n)$ produces a pseudorandom string of length $\binom{t}{k} \ge n^{c'}$. ◀

References

1. Miklos Ajtai, Janos Komlos, and Endre Szemeredi. Deterministic simulation in logspace. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC'87, pages 132–140, New York, NY, USA, 1987. ACM.
2. László Babai, Noam Nisan, and Mario Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. Syst. Sci.*, 45(2):204–232, 1992.
3. Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory of Computing*, 9:283–293, 2013.
4. Mark Braverman, Faith Ellen, Rotem Oshman, Toniann Pitassi, and Vinod Vaikuntanathan. Tight bounds for set disjointness in the message passing model. *CoRR*, abs/1305.4696, 2013.
5. Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. In *FOCS*, pages 40–47, 2010.
6. Joshua Brody and Elad Verbin. The coin problem and pseudorandomness for branching programs. In *FOCS*, pages 30–39, 2010.
7. Amit Chakrabarti. Lower bounds for multi-player pointer jumping. In *IEEE Conference on Computational Complexity*, pages 33–45, 2007.
8. Carsten Damm, Stasys Jukna, and Jiri Sgall. Some bounds on multiparty communication complexity of pointer jumping. *Computational Complexity*, 7(2):109–127, 1998.
9. Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.
10. Yevgeniy Dodis and Daniel Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *STOC*, pages 601–610, 2009.
11. Stefan Dziembowski and Krzysztof Pietrzak. Intrusion-resilient secret sharing. In *FOCS*, pages 227–237, 2007.
12. Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, April 1985.
13. Oded Goldreich and Avi Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Struct. Algorithms*, 11(4):315–343, 1997.
14. Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil P. Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *FOCS*, pages 120–129, 2012.
15. Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *STOC*, pages 356–364, 1994.
16. Richard M. Karp, Christian Schindelhauer, Scott J. Shenker, and Berthold Vocking. Randomized rumor spreading. In *In IEEE Symposium on Foundations of Computer Science*, pages 565–574, 2000.
17. David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'03, pages 482–, Washington, DC, USA, 2003. IEEE Computer Society.
18. Michal Koucky, Prajakta Nimbhorkar, and Pavel Pudlak. Pseudorandom generators for group products. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:113, 2010.
19. Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.
20. Noam Nisan and Amnon Ta-Shma. Extracting randomness: A survey and new constructions. *J. Comput. Syst. Sci.*, 58(1):148–173, 1999.

21 Noam Nisan and David Zuckerman. More deterministic simulation in logspace. In *STOC*, pages 235–244, 1993.

22 Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

23 Ran Raz and Omer Reingold. On recycling the randomness of states in space bounded computation. In *STOC*, pages 159–168, 1999.

24 Omer Reingold, Thomas Steinke, and Salil P. Vadhan. Pseudorandomness for regular branching programs via fourier analysis. *CoRR*, abs/1306.3004, 2013.

25 Michael E. Saks and Shiyu Zhou. $\mathrm{Bp}_h\mathrm{space(s)} \subseteq \mathrm{dspace}(s^{3/2})$. *J. Comput. Syst. Sci.*, 58(2):376–403, 1999.

26 Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.*, 4(2):177–192, 1970.

27 Ronen Shaltiel. Recent developments in explicit constructions of extractors. *Bulletin of the EATCS*, 77:67–95, 2002.

28 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

29 Instructor: Leo Reyzin Scribers: Drew Wolpert and Sophia Yakoubov. Alternating extractors and leakage-resilient stream ciphers. New Developments in Cryptography, MIT, 2011.

A Proof of Lemma 18

Fix $f : \{0,1\}^{rk} \to \{0,1\}$, $\varepsilon > 0$ and $s < {}^{C_\varepsilon(f)}/_k$. Assume towards a contradiction that the i^{th} bit of the output of the generator can be predicted by a non-uniform, space s statistical test. That is, there exists a non-uniform, space s statistical test M^a such that

$$\Pr_{y \in_R \{0,1\}^{tr}}[M^a(\text{first } i-1 \text{ bits of } G(y)) = i^{th} \text{ bit of } G(y)] - \frac{1}{2} > \varepsilon$$

where G is the generator that is defined by the construction above. Fix $y = (y_1, \ldots, y_t) \in (\{0,1\}^r)^t$ and let the i^{th} bit of the output of the generator on input y be $f(S)$, where $S = \{y_{i_1}, y_{i_2}, \ldots, y_{i_k}\}$ and $i_1 > i_2 > \cdots > i_k$. By an averaging argument, we can fix all input strings from y that are not in S, such that the prediction bias of M^a is preserved. We describe a multiparty communication protocol for k players, that computes $f(S)$ with bias ε. The model of this multiparty communication protocol is the NOF model with blackboard communication, in which the j^{th} player knows all input strings except y_{i_j}, for $j \in [k]$, and the players broadcast their messages. The players simulate the running of M^a on the first $i-1$ bits of $G(y)$ as follows. The first player starts the simulation and continues it for as long as she can, that is, as long as she has access to the input bits that the test reads. Then, the first player sends the state of M^a (i.e., all the memory space used by the machine) to the second player. The second player continues the simulation for as long as she can, and so on. Note that for every $j \in [k]$, the j^{th} player can simulate M^a until the simulation requires the value $f(T)$ for a set T that contains y_{i_j}. Moreover, because the sets used to compute the bits of the generator are ordered in anti-lexicographic order, every set that appears after T, until S appears, contains y_{i_j}. Therefore, the k^{th} player can continue the simulation until it reaches a set that contains $y_{i_1}, y_{i_2}, \ldots, y_{i_k}$, which must be the set S, when the simulation ends and the prediction is made. Sending the space used by the machine $k-1$ times, by each of the first $k-1$ players, results in less than ks communicated bits. Since $ks < C_\varepsilon(f)$, we get a contradiction.

On Multiple Input Problems in Property Testing*

Oded Goldreich

Department of Computer Science, Weizmann Institute of Science
Rehovot, Israel
oded.goldreich@weizmann.ac.il

Abstract

We consider three types of multiple input problems in the context of property testing. Specifically, for a property $\Pi \subseteq \{0,1\}^n$, a proximity parameter ϵ, and an integer m, we consider the following problems:

1. Direct m-Sum Problem for Π and ϵ: Given a sequence of m inputs, output a sequence of m bits such that for each $i \in [m]$ the i^{th} bit satisfies the requirements from an ϵ-tester for Π regarding the i^{th} input; that is, for each i, the i^{th} output bit should be 1 (w.p. $\geq 2/3$) if the i^{th} input is in Π, and should be 0 (w.p. $\geq 2/3$) if the i^{th} input is ϵ-far from Π.
2. Direct m-Product Problem for Π and ϵ: Given a sequence of m inputs, output 1 (w.p. $\geq 2/3$) if all inputs are in Π, and output 0 (w.p. $\geq 2/3$) if at least one of the inputs is ϵ-far from Π.
3. The m-Concatenation Problem for Π and ϵ: Here one is required to ϵ-test the m-product of Π; that is, the property $\Pi^m = \{(x_1, \ldots, x_m) : \forall i \in [m]\ x_i \in \Pi\}$.

We show that the query complexity of the first two problems is $\Theta(m)$ times the query complexity of ϵ-testing Π, whereas (except in pathological cases) the query complexity of the third problem is almost of the same order of magnitude as the query complexity of the problem of ϵ-testing Π. All upper bounds are shown via efficient reductions.

We also consider the nonadaptive and one-sided error versions of these problems. The only significant deviation from the picture in the general (adaptive and two-sided error) model is that the *one-sided error* query complexity of the Direct Product Problem equals $\Theta(m)$ times the (two-sided error) query complexity of ϵ-testing Π plus $\Theta(1)$ times the *one-sided error* query complexity of ϵ-testing Π.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes, F.2.2 Complexity Measures and Classes

Keywords and phrases Property Testing, Direct Sum Theorems, Direct Product Theorems, Adaptive vs. Nonadaptive queries, One-Sided Error vs. Two-Sided Error

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.704

1 Introduction

In the last couple of decades, the area of property testing has attracted much attention (see, e. g., a couple of recent surveys [11, 12] and a collection of texts [4]). Loosely speaking, property testing typically refers to sub-linear time probabilistic algorithms for deciding whether a given object has a predetermined property or is far from any object having this property. Such algorithms, called testers, obtain local views of the object by performing queries; that is, the object is seen as a function and the testers get oracle access to this function (and thus may be expected to work in time that is sub-linear in the length of the object).

* This work was partially supported by the Israel Science Foundation (grant No. 1041/08).

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 704–720

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

Indeed, it will be useful to bear in mind that property testing problems are promise problems. The promise is that the object either has the property or is far from having the property, and the task is to distinguish these two cases.

A basic question that was raised in several complexity theoretic contents is that of the complexity of solving several instances of a problem as compared to the complexity of solving a single instance of that problem. Typically, this question appears in two flavors: The first type requires to provide answers to all instances (typically called a direct sum problem), whereas a second type requires to provide a (possibly Boolean) function of these answers (typically called a direct product problem). For example, in the context of Cryptography, the amplification of one-way functions (cf. [2, Sec.2.3]) belongs to the first type, whereas Yao's XOR Lemma (cf. [7]) or the "Selective XOR Lemma" of [6] (cf. [2, Sec. 2.5.2] or [3, Sec. 7.2.1.2]) belongs to the second type.

In this paper, we consider the aforementioned type of questions in the context of property testing. As we shall see, in this context, two natural versions of the direct product problem arise. Hence, we shall consider three types of multiple input problems in the context of property testing, and cast each of them as a promise problem.

1.1 The Three problems

In all cases we refer to a basic property $\Pi \subseteq \{0,1\}^n$ and to a proximity parameter, denoted ϵ, which together determine the promise problem; that is, the basic problem, called ϵ-testing Π, is distinguishing between inputs in Π and inputs in $\overline{\Gamma}_\epsilon(\Pi)$, where $\overline{\Gamma}_\epsilon(\Pi)$ denotes the set of n-bit strings that are ϵ-far from Π (i. e., $x \in \overline{\Gamma}_\epsilon(\Pi)$ iff every $y \in \Pi$ differs from x on at least ϵn bits). (As usual, the ϵ-tester is required to be correct, on each input in $\Pi \cup \overline{\Gamma}_\epsilon(\Pi)$, with probability at least 2/3.) In all cases, we refer to a sequence of m inputs for the ϵ-testing problem.

A Direct Sum Problem. Here the promise is that each of the m inputs is in $\Pi \cup \overline{\Gamma}_\epsilon(\Pi)$, and we are required to solve the basic problem for each of the m inputs. That is, on input $(x_1, \ldots, x_m) \in \{0,1\}^{mn}$, we are required to output $(\sigma_1, \ldots, \sigma_m) \in \{0,1\}^m$ such that for every $i \in [m]$ the following holds:

1. If $x_i \in \Pi$, then $\Pr[\sigma_i = 1] \geq 2/3$.
2. If $x_i \in \overline{\Gamma}_\epsilon(\Pi)$, then $\Pr[\sigma_i = 0] \geq 2/3$.

In other words, we are required to solve m instances of the promise problem, and each instance should be solved with (proximity and error) parameters as in the basic problem.

We show that *the query complexity of the Direct Sum Problem is* $\Theta(m)$ *times the query complexity of* ϵ*-testing* Π. The upper bound is obvious, and our focus is on the lower bound.

A Direct Product Problem. Again, the promise is that each of the m inputs is in $\Pi \cup \overline{\Gamma}_\epsilon(\Pi)$, but here we are required to output the conjunction of the m answers referred to in the Direct Sum Problem. That is, on input $(x_1, \ldots, x_m)$, we should output a bit σ such that the following holds:

1. If $x_i \in \Pi$ holds for each $i \in [m]$, then $\Pr[\sigma = 1] \geq 2/3$.
2. If $x_i \in \overline{\Gamma}_\epsilon(\Pi)$ holds for some $i \in [m]$, then $\Pr[\sigma = 0] \geq 2/3$.

In other words, we are required to distinguish Π^m (i. e., the m-way Cartesian product of Π) from $(\Pi \cup \overline{\Gamma}_\epsilon(\Pi))^m \setminus \Pi^m$. Indeed, this version of a direct product problem is more natural in the current contents than an alternative version that refers to the exclusive or of the aforementioned m answers. In particular, the current direct product problem is related to the concatenation problem that we consider next.

We show that *the query complexity of the Direct Product Problem is* $\Theta(m)$ *times the query complexity of* ϵ*-testing* Π. Note that an upper bound with an $O(m \log m)$ factor is obvious (using error reduction), and something more seems needed in order to to get an $O(m)$ factor.

Indeed, neither the upper bound nor the lower bound is directly related to the corresponding bound for the Direct Sum Problem. In the case of the upper bound, the point is that in the Direct Sum Problem we are only guaranteed that each of the m answers is correct with probability at least $2/3$, whereas the straightforward solution to the Direct Product Problem requires all answers to be correct, which yields an extra $O(\log m)$ factor (for error reduction). The same issue arises when trying to derive the lower bound for the Direct Sum Problem from the one for the Direct Product Problem: Starting from an $\Omega(m)$-factor lower bound for the Direct Product Problem, one only derives an $\Omega(m/\log m)$ factor for the Direct Sum Problem. As hinted above, in both cases, the extra $O(\log m)$ factor can be eliminated (see Lemma 1). We shall use this fact to upper bound the complexity of the Direct Product Problem, but avoid using it for deriving a lower bound for the Direct Sum Problem (because the proof of the lower bound for the Direct Product Problem extends the proof of the lower bound for the Direct Sum Problem).

A Concatenation Problem. Here, the promise is that the sequence of m inputs (or the concatenation of the m inputs) is in $\Pi^m \cup \overline{\Gamma}_\epsilon(\Pi^m)$, where $\overline{\Gamma}_\epsilon(\Pi^m)$ denotes the set of mn-bit strings that are ϵ-far from Π^m, and we are required to distinguish the two cases. That is, on input $(x_1, \ldots, x_m)$, we should output a bit σ such that the following holds:

1. If $x_i \in \Pi$ holds for each $i \in [m]$, then $\Pr[\sigma = 1] \geq 2/3$.
2. If there exists $\epsilon_1, \ldots, \epsilon_m$ that sum-up to $m\epsilon$ such that $x_i \in \overline{\Gamma}_{\epsilon_i}(\Pi)$ holds for each $i \in [m]$, then $\Pr[\sigma = 0] \geq 2/3$.

In other words, we are required to ϵ-test the property Π^m (i. e., the m-way Cartesian product of Π).

We show that *the query complexity of the Concatenation Problem is almost the same as the query complexity of* ϵ*-testing* Π*, provided that the latter increases at least linearly with* $1/\epsilon$, where "almost the same" allows a polylogarithmic slackness factor. Furthermore, if for some $c > 1$ the query complexity of ϵ-testing Π increases at least linearly with $1/\epsilon^c$, then up to a constant factor the query complexity of the Concatenation Problem is the same as the query complexity of ϵ-testing Π. We comment that in all reasonable cases the query complexity of ϵ-testing increases at least linearly with $1/\epsilon$, and an increase rate of at least $1/\epsilon^2$ is very common.

1.2 Nonadaptive Queries and One-sided Error Versions

We also consider the nonadaptive and one-sided error versions of the problems discussed in Section 1.1. The results that we obtain differ from the those obtained in the general model (of adaptive and two-sided error algorithms) in two cases.

Most importantly, it turns out that the *one-sided error* query complexity of the Direct Product Problem equals $\Theta(m)$ times the *two-sided error* query complexity of ϵ-testing Π plus $\Theta(1)$ times the *one-sided error* query complexity of ϵ-testing Π. The point is that the two-sided error query complexity of ϵ-testing Π may be significantly lower than its one-sided error query complexity.[1]

[1] Consider, for example, the set Π of n-bit strings having at least $n/2$ one-entries. Other examples include ρ-clique in the dense graphs model [5], and cycle-freeness in the bounded-degree graph model [8].

The second case refers to the *nonadaptive* query complexity of the Direct Product Problem, where we leave a small gap: We show that the nonadaptive query complexity of the Direct Product Problem is at least $\Omega(m)$ times the nonadaptive query complexity of ϵ-testing Π, and at most $O(m \log m)$ the latter amount.

1.3 Two Comments

Computational Complexity. Our exposition focuses on the query complexity of problems, which is natural since our main focus is on lower bounds. We stress that all our upper bounds are obtained by computationally efficient reductions, yielding computationally efficient algorithms for the multi-instance problems whenever such algorithms are given for the basic testing problem.

A Somewhat Tedious Comment. The Direct Sum and Direct Product problems were stated in Section 1.1 as promise problems regarding a sequence in $(\Pi \cup \overline{\Gamma}_\epsilon(\Pi))^m$. An alternative statement, used in the technical sections, refers to all m-ary sequences and adapts the output requirements accordingly. Specifically, in Section 3 the Direct Sum Problem is defined for all m-ary sequences but requirements are made with respect to the i^{th} answer only when the i^{th} instance is in $\Pi \cup \overline{\Gamma}_\epsilon(\Pi)$. In Section 4 the Direct Product Problem is defined for all m-ary sequences but requirements are made only with respect to sequences that are either in Π^m or contain some instance in $\overline{\Gamma}_\epsilon(\Pi)$. In all cases, our lower bounds hold for the more restricted versions (as stated in Section 1.1) whereas our upper bounds hold also for the relaxed versions (as stated in the technical sections).

1.4 Techniques

Our lower bounds (for the query complexity of the Direct Sum and Product problems) capitalize on the fact that in the context of (computationally unbounded) oracle machines it is easy to decouple the computation on multiple inputs to a sequence of (possibly related) computations on single inputs. In particular, the queries to the different inputs are easily (if not trivially) distinguishable. So the only issue at hand is the allocation of resources (i. e., queries) among the multiple computations; that is, the algorithm solving the multiple-instance problem may allocate its resources in an unequal manner.[2]

In the case that the algorithm solving the multiple-instance problem is nonadaptive (i. e., its queries are determined by its randomness, obliviously of the answers to "prior" queries), we may proceed as follows: First, we identify an index $i \in [m]$ such that the expected number of queries made to the i^{th} instance is at most a $1/m$ fraction of the total number of queries made, denoted q. Next, we truncate executions that make more than $10 \cdot q/m$ queries, obtaining an ϵ-tester for Π that errs with probability at most $(1/3) + 0.1 < 0.45$. Finally, we apply error reduction, and conclude that $O(q/m)$ is lower-bounded by the query complexity of ϵ-testing Π.

However, in general, the algorithm solving the multiple-instance problem may be adaptive (see the proofs of Lemma 1 and Theorem 8 for demonstrations of the possible benefit of adaptivity in the context of multiple-instance problems). In this case the distribution of the algorithm's queries among the m instances may depend on answers to prior queries. We may want to resolve this problem by looking at a "typical" (or "random") sequence of m

[2] Indeed, if the algorithm solving the multiple-instance problem always allocates equal resources to the different instances, then the lower bound follows easily.

instances, but the question is what distribution of instances should we consider. At this point, the MiniMax Principle (cf. Yao [15] following von Neumann [14]) comes handy.

Specifically, our lower bounds for the Direct Sum and Product problems relies on the fact that if the query complexity of ϵ-testing Π is (at least) q, then there exists a distribution of n-bit strings, denoted D, such that every oracle machine M that makes less than q queries errs with probability greater than $1/3$ on the distribution D; that is,

$$\Pr_{x \leftarrow D}\left[(x \in \Pi \wedge M^x = 1) \vee (x \in \overline{\Gamma}_\epsilon(\Pi) \wedge M^x = 0)\right] < 2/3, \tag{1}$$

where $x \leftarrow D$ means that x is selected at random according to D. Indeed, this is the actual contents of Yao's (or von Neumann's) MiniMax Principle.[3]

The lower bound on the Direct Sum Problem is obtained by considering the m-way product of the distribution D. The lower bound on the Direct Product Problem is obtained by considering the m-way product of the distribution D conditioned on having at most one instance in $\overline{\Gamma}_\epsilon(\Pi)$.

The upper bound on the Concatenation Problem uses the fact that the distance in this problem is the average of the distances on the m basic problems. Thus, the tester consists of sampling a few of these basic problems and testing them while using suitable values of the proximity parameter. The economical way in which this is done is inspired by a private communication with Leonid Levin (in mid-1980s).[4]

Regarding the connection between the Direct Sum and Direct Product problems, as noted above, the m-way Direct Product Problem can be easily reduced to the m-way Direct Sum Problem using $O(\log m)$ calls (which are employed for error reduction). Our improved upper bound for the Direct Product Problem is obtained by the following general result, which is implicit in the work of Feige *et al.* [1].[5]

▶ **Lemma 1** (Reducing Direct Product to Direct Sum, Following Thm. 2.7 in [1]). *The m-way direct product problem is reducible to $O(j)$ instances of the $2^{-(j-1)} \cdot m$-way direct sum problem, for every $j = 1, \ldots, \lceil \log_2 m \rceil$.*

Hence, the m-way direct product problem can be solved at the cost of $\sum_{j=1}^{\log_2 m} O(j \cdot 2^{-j} m) = O(m)$ instances of the basic problem. We note that this reduction is quite generic: It holds not only when the basic problem is ϵ-testing some property, but rather with respect to any randomized procedures for solving decision problems with constant error probability.

Proof. The issue is that when we invoke the direct sum algorithm on a sequence of m instances for the basic problem Π, we may get many 0-answers even if all instances are in Π (and so we cannot distinguish this case from the case in which the sequence of instances contains few instances in $\overline{\Gamma}_\epsilon(\Pi)$). Our solution is to declare the instances for which a 0-answer

---

[3] Let us mention, in passing, that we have always objected to the practice of attributing the converse of the above to Yao [15] (or to von Neumann [14]). That is, the fact that the existence of D such that Eq. (1) holds implies that the query complexity of ϵ-testing Π is at least q is a triviality. What is non-trivial is the fact that this method of obtaining lower bounds is actually "complete" (i. e., it yields the best possible lower bounds). This non-trivial direction is the one we use here. For further discussion, see Appendix A.1.

[4] The idea appeared in [9, Sec. 9], and we do not recall a prior use of it. Following [9], this idea was also used in [6, Lem. 3] (see also [2, Clm. 2.5.4.1]). Within the context of property testing, this idea was first used in [8] (see Lemma 3.3 in the proceeding version and Lemma 3.6 in the journal version). For further discussion see Appendix A.2.

[5] The result of Feige *et al.* [1] is stated in terms of computing an m-wise AND by a noisy decision tree. A simpler procedure is provided by Newman [10, Obs. 2.2]. Our procedure is different from both.

was obtained as candidates for being in $\overline{\Gamma}_\epsilon(\Pi)$, and apply the direct sum algorithm only to the surviving candidates. This process is iterated, as long as the set of candidates is not too big, which may happen only if this set contains many instances in $\overline{\Gamma}_\epsilon(\Pi)$. For this process to work, we reduce the error probability in the various iterations such that in the j^{th} iteration we use $O(j)$ repetitions and have error probability 2^{-j} (or so) per each instance. This implies that the set of false candidates (i. e., candidates that are actually in Π) does shrink in each iteration, while each instance in $\overline{\Gamma}_\epsilon(\Pi)$ survives all iterations with high probability (e. g., with probability at least 0.9). Details follow.

Given an instance $(x_1, \ldots, x_m)$ for the direct product problem, we proceed in $\ell = \lceil \log_2(3m) \rceil$ iterations, while maintaining a set $I \subseteq [m]$ of candidates (for being in $\overline{\Gamma}_\epsilon(\Pi)$). Initially, $I = [m]$. In the j^{th} iteration, if $|I| > 2^{-(j-1)} \cdot m$, then we output 0. Otherwise, we invoke the $|I|$-way direct sum algorithm on $(x_{i_1}, \ldots, x_{i_t})$, where $I = \{i_1, \ldots, i_t\}$, for $O(j)$ times, and rule by majority on each of the x_i's (with $i \in I$). We keep i in I if the majority vote on x_i is 0. If, at the end of any iteration, the set I becomes empty, then we output 1.

Specifically, in the j^{th} iteration, the direct sum algorithm is invoked for $O(j)$ times in order to guarantee that the majority vote (on each x_i) is correct with probability at least $1 - 2^{-(j+3)}$. Hence, each $x_i \in \overline{\Gamma}_\epsilon(\Pi)$ has a fair chance to survive all iterations, in which case I will become too big at some iteration (since $2^{-(\ell-1)} < 1/m$). On the other hand, with very high probability, the vote on almost all inputs is correct. Hence, if all x_i's are in Π, then we expect the set I to be cut by a factor of at least two in each iteration, and so eventually $I = \emptyset$. Further details follow.

We first note that if any of the x_i's is in $\overline{\Gamma}_\epsilon(\Pi)$, then i remains in I throughout all iterations, with probability at least $1 - \sum_{j=1}^{\ell} 2^{-(j+3)} > 7/8$. In this case the algorithm outputs 0, because, for some $j \geq 2$ (possibly for $j = \ell$), at the beginning of the j^{th} iteration it holds that $|I| > 2^{-(j-1)} \cdot m$.

On the other hand, if $(x_1, \ldots, x_m) \in \Pi^m$, then with probability at least $1 - \sum_{j=1}^{\ell} 2^{-(j+2)} > 3/4$ in each iteration j it holds that $|I| \leq 2^{-(j-1)} \cdot m$. (This is the case since the expected size of I is cut by a factor of $2^{-(j+3)}$, and so with probability at least $1 - 2^{-(j+2)}$ it is cut by half.) In this case the algorithm outputs 1, because for some $j \geq 2$ (possibly for $j = \ell$) at the end of the j^{th} iteration it holds that I is empty. ◀

1.5 Organization

Following a short preliminaries section, we proceed to the study of the three problems described above: The Direct Sum Problem is studied in Section 3, the Direct Product Problem is studied in Section 4, and the Concatenation Problem is studied in Section 5. In Section 6 we consider nonadaptive and one-sided error versions of these problems.

In the appendix, we elaborate on two comments that were made in Section 1.4 (see Footnotes 3 and 4): Section A.1 discusses Yao's MiniMax Principle, whereas Section A.2 surveys a general method (which we call Levin's Economical Work Investment Strategy) that underlies some of the saving obtained in Section 5.

2 Preliminaries

For sake of simplicity, we present all results in terms of properties of fixed-length strings (i. e., n is fixed), and while referring to testing them with respect to a fixed value of the proximity parameter, denoted ϵ. Nevertheless, both parameters should be viewed as generic (and thus varying).

▶ **Definition 2** (Property Testing). Let $\Pi \subseteq \{0,1\}^n$ and $\epsilon > 0$. An ϵ-tester for Π is a randomized oracle machine T that satisfies the following two conditions.

1. If $x \in \Pi$, then $\Pr[T^x = 1] \geq 2/3$.
2. If $x \in \{0,1\}^n$ is ϵ-far from Π, then $\Pr[T^x = 0] \geq 2/3$, where the distance between strings is the fraction of bits on which they disagree and the distance to a set is the distance to the closest element in the set. That is, x is ϵ-far from Π if and only if every $y \in \Pi$ differs from x on at least ϵn bits.

The query complexity of T is the maximum number of queries that T makes, when the maximization is over all $x \in \{0,1\}^n$ and over the coin tosses of T. The query complexity of ϵ-testing Π, denoted $\mathtt{Q}_\epsilon(\Pi)$, is the minimum query complexity of all ϵ-testers for Π.

Indeed, ϵ-testing is the promise problem that consists of distinguishing inputs in Π from inputs that are ϵ-far from Π.

3 The Direct Sum Problem

Recall that in this problem, on input $(x_1, \ldots, x_m) \in \{0,1\}^{mn}$, we should output $(\sigma_1, \ldots, \sigma_m) \in \{0,1\}^m$ such that for every $i \in [m]$ the following holds:

1. If $x_i \in \Pi$, then $\Pr[\sigma_i = 1] \geq 2/3$.
2. If $x_i \in \overline{\Gamma}_\epsilon(\Pi)$, then $\Pr[\sigma_i = 0] \geq 2/3$.

Let us denote this problem by $\mathtt{DS}^m_\epsilon(\Pi)$.

▶ **Theorem 3** (The Direct Sum Theorem). *For every property Π, proximity parameter ϵ, and integer m, the query complexity of $\mathtt{DS}^m_\epsilon(\Pi)$ is $\Theta(m \cdot \mathtt{Q}_\epsilon(\Pi))$.*

Here and in all our results the hidden constants in the Θ notation are universal (i. e., are independent of Π, ϵ and m). Ditto for the O and Ω notations.

Proof. The upper bound holds by merely invoking the ϵ-tester, T, of Π for m times; that is, on input $(x_1, \ldots, x_m)$, we output $(T^{x_1}, \ldots, T^{x_m})$.

Turning to the lower bound, we start by invoking the MiniMax Principle of von Neumann [14] as adapted by Yao [15]. That is, let $q \stackrel{\text{def}}{=} \mathtt{Q}_\epsilon(\Pi)$, and consider a two-player zero-sum game between an algorithmic player and an adversarial player. In the game, the algorithmic player selects (randomly or deterministically) a deterministic oracle machine M that makes at most $q-1$ queries and the adversarial player selects (randomly or deterministically) an input x. The algorithmic player wins if and only if either $x \notin (\Pi \cup \overline{\Gamma}_\epsilon(\Pi))$ or M^x outputs the correct answer.

By our hypothesis, the algorithmic player has no strategy that guarantees winning with probability at least $2/3$; that is, for every distribution on deterministic oracle machines $\mathcal{M}$ that make at most $q-1$ queries, there exists a string $x \in \Pi \cup \overline{\Gamma}_\epsilon(\Pi)$ such that when selecting $M \leftarrow \mathcal{M}$ the probability that M^x is correct is smaller than $2/3$. The MiniMax Principle asserts that, in this case, there exists a distribution of inputs in $\Pi \cup \overline{\Gamma}_\epsilon(\Pi)$ on which each (deterministic) oracle machine M that make at most $q-1$ queries fails with probability greater than $1/3$. Let us denote this distribution by D.

Now, consider an arbitrary (randomized) oracle machine $\overline{M}_0$ of query complexity q_0 that solves $\mathtt{DS}^m_\epsilon(\Pi)$. By straightforward amplification, we obtain a machine $\overline{M}$ of query complexity $q_1 = 10q_0$ having an error probability of at most $1/6$ on each answer. That is, letting $\overline{M}_i^{\overline{x}}$ denote the i^{th} bit in $\overline{M}^{\overline{x}}$, for each $\overline{x} = (x_1, \ldots, x_m)$ and each $i \in [m]$, the following holds:

1. If $x_i \in \Pi$, then $\Pr[\overline{M}_i^{\overline{x}} = 1] \geq 5/6$.
2. If $x_i \in \overline{\Gamma}_\epsilon(\Pi)$, then $\Pr[\overline{M}_i^{\overline{x}} = 0] \geq 5/6$.

Now, consider the m-way Cartesian product of D, denoted D^m, and consider the execution of $\overline{M}$ on an input drawn from D^m. Fix $i \in [m]$ such that the expected number of queries that $\overline{M}$ makes to its i^{th} input is at most $q' = q_1/m$, where the expectation is taken over D^m as well as over the coins of $\overline{M}$. Then, with probability at least $5/6$, machine $\overline{M}$ makes at most $6q'$ queries to its i^{th} input. It follows that, with probability at least $(5/6) - (1/6) = 2/3$, machine $\overline{M}$ makes at most $6q'$ queries to its i^{th} input and yet answers correctly regarding this input.

Using $\overline{M}$, we obtain a randomized oracle machine M' that makes at most $6q'$ queries and solves the basic problem on D with probability at least $2/3$, where the probability is taken over both D and the internal coins of M': Machine M' just emulates the execution of $\overline{M}$, while using its own input as the i^{th} input of $\overline{M}$, emulating the other inputs (by generating $m-1$ samples according to D), and terminating the execution of $\overline{M}$ if $\overline{M}$ tries to make more than $6q'$ queries to its i^{th} input (which happens with probability at most $1/6$). Using an averaging argument, we obtain a deterministic oracle machine M'' that makes at most $6q'$ queries and succeeds on D with probability at least $2/3$. It follows that $6q' > q - 1$ (or $6q' \geq q$), and thus $q_0 = \frac{q_1}{10} = \frac{m \cdot q'}{10} \geq \frac{m \cdot q}{60}$. ◀

4 The Direct Product Problem

Recall that in this problem, on input $(x_1, \ldots, x_m) \in \{0,1\}^{mn}$, we should output $\sigma \in \{0,1\}$ such that the following holds:

1. If for every $i \in [m]$ it holds that $x_i \in \Pi$, then $\Pr[\sigma = 1] \geq 2/3$.
2. If there exists $i \in [m]$ such that $x_i \in \overline{\Gamma}_\epsilon(\Pi)$, then $\Pr[\sigma = 0] \geq 2/3$.

Let us denote this problem by $\mathtt{DP}^m_\epsilon(\Pi)$.

▶ **Theorem 4** (The Direct Product Theorem). *For every property Π, proximity parameter ϵ, and integer m, the query complexity of $\mathtt{DP}^m_\epsilon(\Pi)$ is $\Theta(m \cdot \mathtt{Q}_\epsilon(\Pi))$.*

Proof. The upper bound follows by combining Lemma 1 with the upper bound of Theorem 3: That is, we use the reduction of the lemma, and apply the straightforward algorithm asserted by the theorem. Turning to the lower bound, we start as in the proof of Theorem 3, and let D denote the corresponding "hard" distribution. Actually, we consider ϵ-testers for Π that are (only) correct with probability at least 0.51, and derive a lower bound of $\Omega(\mathtt{Q}_\epsilon(\Pi))$ on their query complexity, denoted q, because otherwise a contradiction follows by a straightforward amplification.

Let D' denote the distribution obtained from D by conditioning that the string is in Π, and likewise D'' is obtained by conditioning that the string is in $\overline{\Gamma}_\epsilon(\Pi)$. For each $i \in [m]$, we will consider the distribution $D^{(i)}$ that consists of the Cartesian product of $m-1$ copies of D' and a single copy of D'' placed in the i^{th} position. We shall also consider the m-way Cartesian product of D', denoted $(D')^m$.

Given any oracle machine $\overline{M}_0$ of query complexity q_0 that solves $\mathtt{DP}^m_\epsilon(\Pi)$, we consider its amplification to a machine $\overline{M}$ of query complexity $q_1 = O(q_0)$ having error probability at most 0.01 on each input. We first consider the invocation of $\overline{M}$ on inputs drawn from $(D')^m$, and fix $i \in [m]$ such that the expected number of queries that $\overline{M}$ makes to its i^{th} input is at most $q' = q_1/m$. Hence, with probability at least 0.99, machine $\overline{M}$ makes at most $100q'$ queries to the i^{th} input. We now consider two cases regarding the execution of $\overline{M}$ on an input drawn from $D^{(i)}$:

1. If when invoked on $D^{(i)}$, with probability at least $5/6$, machine $\overline{M}$ makes at most $100q'$ queries to the i^{th} input, then we proceed as in the proof of Theorem 3. Specifically, in

this case the $100q'$-query truncated executions of $\overline{M}$ yield a machine M'' for ϵ-testing Π, and so $100q' > q - 1$ must hold (which in turn implies $q_0 = \Omega(m \cdot q)$).

2. Otherwise (with probability at least $1/6$, machine $\overline{M}$ makes more than $100q'$ queries to the i^{th} input), we can use this as an indication to whether $\overline{M}$ runs on $(D')^m$ or on $D^{(i)}$, which in turn yields an ϵ-tester for Π. Specifically, consider an alternative randomized machine T that on input $x \in \{0,1\}^n$ invokes $\overline{M}$ on input $(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_m)$, where $(x_1, \ldots, x_m) \leftarrow (D')^m$, outputs 0 if $\overline{M}$ tries to make more than $100q'$ queries to x, and otherwise outputs the outcome of a coin with bias 0.55 towards 1. Then, $\Pr_{x \leftarrow D'}[T^x = 1] \geq 0.99 \cdot 0.55 > 0.51$, whereas $\Pr_{x \leftarrow D''}[T^x = 0] \geq \frac{5}{6} \cdot 0.45 + \frac{1}{6} > 0.51$. Thus, the probability that T decides D correctly exceeds 0.51, and we again derive $100q' > q - 1$.

The theorem follows. ◀

5 The Concatenation Problem

Recall that in this problem, on input $(x_1, \ldots, x_m) \in \{0,1\}^{mn}$, we should output $\sigma \in \{0,1\}$ such that the following holds:

1. If for every $i \in [m]$ it holds that $x_i \in \Pi$ (equiv., $(x_1, \ldots, x_m) \in \Pi^m$), then $\Pr[\sigma = 1] \geq 2/3$.
2. If there exists $\epsilon_1, \ldots, \epsilon_m$ that sum-up to $m\epsilon$ such that for every $i \in [m]$ it holds that $x_i \in \overline{\Gamma}_{\epsilon_i}(\Pi)$ (equiv., $(x_1, \ldots, x_m) \in \overline{\Gamma}_{\epsilon}(\Pi^m)$), then $\Pr[\sigma = 0] \geq 2/3$.

Let us denote this problem by $\mathtt{CP}^m_\epsilon(\Pi)$.

▶ **Theorem 5** (The First Concatenation Theorem). *Suppose that $\mathtt{Q}_\epsilon(\Pi)$ increases at least linearly with $1/\epsilon$; that is, $\mathtt{Q}_{\epsilon/2}(\Pi) \geq \min(\Omega(n), 2 \cdot \mathtt{Q}_\epsilon(\Pi))$ for every $\epsilon > 0$. Then, the query complexity of $\mathtt{CP}^m_\epsilon(\Pi)$ is $O(\log(1/\epsilon))^3 \cdot \mathtt{Q}_{\epsilon/4}(\Pi) = \widetilde{O}(\mathtt{Q}_{\epsilon/4}(\Pi))$.*

The condition $\mathtt{Q}_{\epsilon/2}(\Pi) \geq \min(\Omega(n), 2 \cdot \mathtt{Q}_\epsilon(\Pi))$ must be made to avoid making the requirement an impossible one, since any property Π can be tested with n queries. Further relaxations of the condition are possible, but the conclusion should be corrected accordingly. For example, if $\mathtt{Q}_{\epsilon/2}(\Pi) \geq 2 \cdot \mathtt{Q}_\epsilon(\Pi)$ holds for every $\epsilon \geq \epsilon_0(n)$, then the conclusion holds for every $\epsilon \geq 2\epsilon_0(n)$.

Proof. Fixing any $(x_1, \ldots, x_m)$, let $\delta_i \in [0,1]$ denote the distance of x_i from Π. The key observation is that if $\sum_{i \in [m]} \delta_i \geq m \cdot \epsilon$, then for $\ell = \lceil \log_2(1/\epsilon) \rceil + 1$ there exists some $j \in [\ell]$ such that $|\{i \in [m] : \delta_i \in [2^{-j}, 2^{-(j-1)}]\}| \geq m \cdot 2^j\epsilon/4\ell$ (see Fact A.1).[6] But in such a case, sampling $O(\ell/2^j\epsilon)$ indices i, and 2^{-j}-testing each input x_i w.r.t Π will do (i. e., with high constant probability, we will sample an instance that is 2^{-j}-far from Π). Details follow.

First, let us spell out the proposed ϵ-tester (for Π^m). On input $(x_1, \ldots, x_m)$, for every $j \in [\ell]$, the tester samples $O(\ell/2^j\epsilon)$ indices i, and 2^{-j}-tests each input x_i w.r.t Π, with error probability ϵ^2 (rather than 1/3).[7] Hence, we make $O(\ell) \cdot \mathtt{Q}_{2^{-j}}(\Pi)$ queries to each x_i that is sampled in the j^{th} iteration, and can neglect the probability that we obtained a wrong result for any x_i in any iteration. The tester accepts if and only if all the aforementioned tests answer 1 (i. e., all sampled x_i's were verified as being in Π).

Turning to the analysis of the above tester, we first note that the number of queries made by it is upper-bounded by $q \stackrel{\text{def}}{=} \sum_{j \in [\ell]} O(\ell/2^j\epsilon) \cdot O(\ell \cdot \mathtt{Q}_{2^{-j}}(\Pi))$, whereas $\mathtt{Q}_{2^{-j}}(\Pi) \leq \frac{\epsilon/4}{2^{-j}} \cdot \mathtt{Q}_{\epsilon/4}(\Pi)$ (by the theorem's hypothesis). Thus, $q \leq \sum_{j \in [\ell]} O(\ell^2) \cdot \mathtt{Q}_{\epsilon/4}(\Pi)$, which meets the asserted

[6] Consider the uniform distribution over $[m]$, and let $q(s) = \delta_s$.

[7] Error reduction is used here in order to upper bound the probability that any of the tests returns 0 when $(x_1, \ldots, x_m) \in \Pi^m$. Indeed, this is not necessary in the case of one-sided error. Actually, one can avoid the error reduction step also in the case of two-sided error by using the ideas underlying Lemma 1 (and save a factor of $O(\ell)$), but we did not bother to do so.

complexity bound. Next, note that if $(x_1,\ldots,x_m) \in \Pi^m$, then each of the tests answers 1 with probability at least $1-\epsilon^2$, whereas the number of tests is $\sum_{j\in[\ell]} O(\ell/2^j\epsilon) = O(\ell/\epsilon)$. On the other hand, if $(x_1,\ldots,x_m) \in \overline{\Gamma}_\epsilon(\Pi^m)$, then, for some $j\in[\ell]$, with probability at least $3/4$, some $x_i \in \overline{\Gamma}_{2^{-j}}(\Pi)$ is sampled and 2^{-j}-tested (and so answered 0 with probability at least $1-\epsilon^2 > 0.99$). The theorem follows. ◀

▶ **Theorem 6** (The Second Concatenation Theorem). *Suppose that, for some constant $c>1$, it holds that $\mathtt{Q}_\epsilon(\Pi)$ increases at least linearly with $1/\epsilon^c$; that is, $\mathtt{Q}_{\epsilon/2}(\Pi) \geq \min(\Omega(n), 2^c\cdot\mathtt{Q}_\epsilon(\Pi))$ for every $\epsilon>0$. Then, the query complexity of $\mathtt{CP}^m_\epsilon(\Pi)$ is $O(\mathtt{Q}_{\epsilon/4}(\Pi))$.*

Proof. We follow the proof of Theorem 5, while using a slightly different analysis of the various "buckets" (or the sets) $B_j \stackrel{\text{def}}{=} \{i\in[m] : \delta_i \in [2^{-j},2^{-(j-1)}]\}$. Specifically, for $\ell = \lceil \log_2(1/\epsilon)\rceil + 1$ and $p(j) \stackrel{\text{def}}{=} (\ell+5-j)^{-2}$, we first prove that *if* $\sum_{i\in[m]}\delta_i \geq m\cdot\epsilon$, *then there exists some* $j\in[\ell]$ *such that* $|B_j| \geq m\cdot p(j)\cdot 2^j\epsilon$. This is essentially proved in Fact A.2, and the argument is adapted next (for the reader's convenience). Indeed, assuming towards the contradiction that for every $j\in[\ell]$ it holds that $|B_j| < m\cdot p(j)\cdot 2^j\epsilon$, we get

$$\begin{aligned}\sum_{i\in[m]}\delta_i &< \left(\sum_{j\in[\ell]} m\cdot p(j)\cdot 2^j\epsilon\cdot 2^{-(j-1)}\right) + m\cdot 2^{-\ell}\\ &\leq 2m\epsilon\cdot\left(\sum_{j\in[\ell]} p(j)\right) + m\cdot\epsilon/2\\ &< 2\cdot\frac{m\epsilon}{2}\end{aligned}$$

where the last inequality uses $\sum_{j\in[\ell]} p(j) < 1/4$. But this contradicts the hypothesis that $\sum_{i\in[m]}\delta_i \geq m\epsilon$.

We next present the modified ϵ-tester. On input $(x_1,\ldots,x_m)$, for every $j\in[\ell]$, the tester samples $O((p(j)\cdot 2^j\epsilon)^{-1})$ indices i, and 2^{-j}-tests each input x_i w.r.t Π with error probability $p(j)^2\cdot 2^j\epsilon/O(1) = \exp(-\Theta(\ell+1-j))$. Hence, we make $O(\ell+1-j)\cdot\mathtt{Q}_{2^{-j}}(\Pi)$ queries to each x_i that is sampled in the j^{th} iteration, and can neglect the the probability that we obtained a wrong result for any x_i in any iteration (since a wrong answer is obtained in the j^{th} iteration with probability at most $p(j)/O(1)$). The number of queries made by this tester is upper-bounded by

$$\begin{aligned}&\sum_{j\in[\ell]} O((p(j)\cdot 2^j\epsilon)^{-1})\cdot O(\ell+1-j)\cdot\mathtt{Q}_{2^{-j}}(\Pi)\\ &\qquad\leq O(1/\epsilon)\cdot\sum_{j\in[\ell]}\frac{\ell+5-j}{p(j)2^j}\cdot\left(\frac{\epsilon/4}{2^{-j}}\right)^c\cdot\mathtt{Q}_{\epsilon/4}(\Pi)\\ &\qquad= O(\epsilon^{c-1})\cdot\sum_{k\in[\ell]}(k+4)^3\cdot 2^{(c-1)(\ell+1-k)}\cdot\mathtt{Q}_{\epsilon/4}(\Pi)\\ &\qquad\leq O(1)\cdot\sum_{k\in[\ell]}(k+4)^3\cdot 2^{-(c-1)k}\cdot\mathtt{Q}_{\epsilon/4}(\Pi)\end{aligned}$$

where the first inequality uses the theorem's hypothesis, the equality uses the definition of $p(j)$ (and the substitution $k=\ell+1-j$), and the last inequality uses $\ell = \lceil\log_2(1/\epsilon)\rceil+1 < \log_2(4/\epsilon)$. Using $\sum_{k\in[\ell]}(k+4)^3\cdot 2^{-c'k} = O(1)$, for any $c'>0$, the claim follows. ◀

6 Ramifications: Nonadaptivity and One-Sided Error

In this section, we consider the ramification of our study to nonadaptive algorithms and to one-sided error algorithms. Specifically, in Section 6.1 we consider nonadaptive algorithms (with two-sided error), whereas in Section 6.2 we consider (adaptive) one-sided error algorithms. In each of these cases, we compare the complexity of restricted algorithms solving the multiple-instance problems to the complexity of similarly restricted ϵ-testers of Π.

We note that nonadaptivity "dominates" one-sided error (see Section 6.3): The results for nonadaptive algorithms with one-sided error behave more like the results for nonadaptive algorithms with two-sided error than the results for general (i. e., adaptive) algorithms with one-sided error.

6.1 Nonadaptive Algorithms

An algorithm (or rather an oracle machine) is called nonadaptive if it determines its queries solely based on its randomness, regardless of the answers obtained to "prior" queries. (Indeed, in this case, the order of the queries is arbitrary and/or immaterial.)

As noted at the beginning of Section 1.4, the lower bounds for the Direct Sum and Product problems are much easier to establish in the nonadaptive case. In this case, the queries of the algorithm solving the multi-instance problem are oblivious of the instances, and so we can easily derive from it a single-instance algorithm (of easily related query complexity). On the other hand, the efficient reduction of the Direct Product Problem to the Direct Sum Problem, captured by Lemma 1, does not seem to work anymore (since the reduction that we presented is inherently adaptive). Letting $\mathtt{Q}^{\mathrm{na}}_\epsilon$ denote the minimum query complexity of all nonadaptive ϵ-testers for Π, we get:

▶ **Theorem 7** (Nonadaptive Query Complexity of Multiple-instance Problems)**.** *For every property Π, proximity parameter ϵ, and integer m, the following holds.*

The Direct Sum Problem: *The nonadaptive query complexity of $\mathtt{DS}^m_\epsilon(\Pi)$ is $\Theta(m \cdot \mathtt{Q}^{\mathrm{na}}_\epsilon(\Pi))$.*

The Direct Product Problem: *The nonadaptive query complexity of $\mathtt{DP}^m_\epsilon(\Pi)$ is $\Omega(m) \cdot \mathtt{Q}^{\mathrm{na}}_\epsilon(\Pi)$ and $O(m \log m) \cdot \mathtt{Q}^{\mathrm{na}}_\epsilon(\Pi)$.*

The Concatenation Problem: *For $c \geq 1$, suppose that $\mathtt{Q}_\epsilon(\Pi)$ increases at least linearly with $1/\epsilon^c$. Then, the nonadaptive query complexity of $\mathtt{CP}^m_\epsilon(\Pi)$ is $O(\log(1/\epsilon))^3 \cdot \mathtt{Q}^{\mathrm{na}}_{\epsilon/4}(\Pi)$ if $c = 1$, and $O(\mathtt{Q}^{\mathrm{na}}_{\epsilon/4}(\Pi))$ otherwise* (i. e., for $c > 1$).

Indeed, closing the gap for the Direct Product Problem is left as an open problem. As observed by Ron Rothblum, the gap disappears whenever $\mathtt{Q}^{\mathrm{na}}_\epsilon(\Pi) = O(\mathtt{Q}^{\mathrm{dr}}_\epsilon(\Pi))$, where $\mathtt{Q}^{\mathrm{dr}}_\epsilon(\Pi)$ is as defined in Section 6.3. Note that it may be the case that the nonadaptive query complexity of $\mathtt{DP}^m_\epsilon(\Pi)$ is $\Theta(m) \cdot \mathtt{Q}^{\mathrm{na}}_\epsilon(\Pi)$ for some Π and $\Theta(m \log m) \cdot \mathtt{Q}^{\mathrm{na}}_\epsilon(\Pi)$ for others.

Proof Sketch. Both lower bounds follow by starting with an algorithm that solves the multi-instance problem with query complexity q, and deriving an ϵ-tester for Π with *expected* query complexity of at most q/m. By truncating executions that make more than $10q/m$ queries, we obtain an ϵ-tester that errs with probability $(1/3) + 0.1$. Lastly, error reduction yields a standard ϵ-tester for Π of query complexity $O(q/m) \geq \mathtt{Q}_\epsilon(\Pi)$, and the lower bound claims follow (for both problems). Regarding the Concatenation Problem, we note that the reductions presented in Section 5 are actually nonadaptive, and so the upper bound claims follow. ◀

6.2 One-sided Error Algorithms

An ϵ-tester for Π is said to have one-sided error if it always accepts (i. e., output 1) inputs in Π (rather than accept them with probability at least 2/3). We denote by $\mathtt{Q}^{\mathrm{ose}}_{\epsilon}$ the minimum query complexity of all one-sided error ϵ-testers for Π. The notion of one-sided error algorithms extends naturally to the three multiple-instance problems we have studied: In the Direct Sum Problem it asserts that σ_i is always 1 if $x_i \in \Pi$, whereas in the other two problems it asserts that the algorithm always accepts a sequence in Π^m.

In the context of one-sided error algorithms, there is a simpler reduction of the Direct Product Problem to the Direct Sum Problem (i. e., simpler than the one captured by Lemma 1); that is, solving $\mathtt{DP}^m_{\epsilon}(\Pi)$ (with one-sided error) reduces to a single invocation of a (one-sided error) algorithm solving $\mathtt{DS}^m_{\epsilon}(\Pi)$. The Direct Sum solver outputs 1 if and only if the m-bit long vector of answers obtained for the Direct Sum Problem is all 1. This implies that *the one-sided error query complexity of* $\mathtt{DP}^m_{\epsilon}(\Pi)$ *is* $O(m \cdot \mathtt{Q}^{\mathrm{ose}}_{\epsilon}(\Pi))$, but we shall see a stronger result below. Regarding the Concatenation Problem, as in the nonadaptive case, it is clear that the algorithms presented in Section 5 operate also in the case of one-sided error (i. e., the reductions preserve one-sided error). Actually, we can save a $\log(1/\epsilon)$ factor (see Footnote 7).

Regarding the lower bounds, it turns out that the argument for the Direct Sum Problem can be adapted (as shown below), but the one for Direct Product Problem fails. In fact, there are cases in which the one-sided error query complexity of the Direct Product Problem is very close to the one-sided error query complexity of testing Π. Specifically, the one-sided error query complexity of $\mathtt{DP}^m_{\epsilon}(\Pi)$ is $O(m \cdot \mathtt{Q}_{\epsilon}(\Pi) + \mathtt{Q}^{\mathrm{ose}}_{\epsilon}(\Pi))$, whereas in some cases $\mathtt{Q}_{\epsilon}(\Pi)$ is much smaller than $\mathtt{Q}^{\mathrm{ose}}_{\epsilon}(\Pi)$ (e. g., ϵ-testing ρ-clique in the dense graphs model has poly$(1/\epsilon)$-query two-sided error tester [5, Sec. 7], but no $o(\sqrt{n})$-query one-sided error testers [5, sec. 10.1.6]).[8]

▶ **Theorem 8** (One-sided Error Query Complexity of Multiple-instance Problems). *For every property Π, proximity parameter ϵ, and integer m, the following holds.*

The Direct Sum Problem: *The one-sided error query complexity of* $\mathtt{DS}^m_{\epsilon}(\Pi)$ *is* $\Theta(m \cdot \mathtt{Q}^{\mathrm{ose}}_{\epsilon}(\Pi))$.

The Direct Product Problem: *The one-sided error query complexity of* $\mathtt{DP}^m_{\epsilon}(\Pi)$ *is* $\Theta(m \cdot \mathtt{Q}_{\epsilon}(\Pi) + \mathtt{Q}^{\mathrm{ose}}_{\epsilon}(\Pi))$.

The Concatenation Problem: *For $c \geq 1$, suppose that* $\mathtt{Q}_{\epsilon}(\Pi)$ *increases at least linearly with* $1/\epsilon^c$. *Then, the one-sided error query complexity of* $\mathtt{CP}^m_{\epsilon}(\Pi)$ *is* $O(\log(1/\epsilon))^2 \cdot \mathtt{Q}^{\mathrm{ose}}_{\epsilon/4}(\Pi)$ *if $c = 1$, and* $O(\mathtt{Q}^{\mathrm{ose}}_{\epsilon/4}(\Pi))$ *otherwise* (i. e., for $c > 1$).

Proof Sketch. When proving the lower bound for $\mathtt{DS}^m_{\epsilon}(\Pi)$, we modify the distribution D as follows. For $q = \mathtt{Q}^{\mathrm{ose}}_{\epsilon}(\Pi)$, the algorithmic player we consider here selects (randomly or deterministically) a deterministic oracle machine that makes at most $q-1$ queries and *always outputs 1 on any input in* Π. As before, the adversarial player selects (randomly or deterministically) an input in $\Pi \cup \overline{\Gamma}_{\epsilon}(\Pi)$. Now, D is a distribution on inputs in $\Pi \cup \overline{\Gamma}_{\epsilon}(\Pi)$ such that for any deterministic machine M of query complexity at most $q-1$ that always accepts inputs in Π it holds that $\Pr_{x \in D}[M^x = 0 | x \in \overline{\Gamma}_{\epsilon}(\Pi)] < 2/3$ (which in particular means that $\Pr_{x \in D}[x \in \overline{\Gamma}_{\epsilon}(\Pi)] > 0$). Finally, we proceed as in the proof of Theorem 3, with two exceptions:

1. Here we start with a one-sided error machine $\overline{M}_0$ that solves the multi-instance problem $\mathtt{DS}^m_{\epsilon}(\Pi)$, and our aim is to derive a one-sided error ϵ-tester for Π.

[8] Another natural example is ϵ-testing cycle-freeness in the bounded-degree graphs model [8, Sec. 4]. Also note that the set Π of n-bit strings having at least $n/2$ one-entries has a two-sided error $O(1/\epsilon^2)$-query tester, but requires at least $n/2$ queries for one-sided error testing.

2. When truncating executions that make too many queries to the i^{th} instance, we make the modified algorithm output 1.

   Note that in the proof of Theorem 3, the output in these truncated executions was left unspecified. Hence, the specific setting used above does not affect the validity of the analysis of the case that the instance is in $\overline{\Gamma}_\epsilon(\Pi)$. But this specific setting guarantees that inputs in Π are always accepted (by the single-instance algorithm that we derive).

This completes the proof of the lower bound for $\mathtt{DS}^m_\epsilon(\Pi)$.

Turning to $\mathtt{DP}^m_\epsilon(\Pi)$, we first note that the lower bound follows by combing the lower bound on two-sided testers (i. e., $\Omega(m \cdot \mathtt{Q}_\epsilon(\Pi))$) with the obvious lower bound of $\mathtt{Q}^{\text{ose}}_\epsilon(\Pi)$. As for the upper bound, we consider an algorithm that first finds a candidate instance in $\overline{\Gamma}_\epsilon(\Pi)$, by using a two-sided error ϵ-tester, and then applies a one-sided error ϵ-tester to it. The first step is easy to implement by invoking the two-sided error tester $O(\log m)$ times on each instance (and ruling by majority), but we aim at a better upper bound. The idea is to use a modification of the algorithm presented in the proof of Lemma 1. Specifically, on input $(x_1, \ldots, x_m)$, the algorithm, denoted A, proceeds as follows:

1. Algorithm A invokes the reduction presented in the proof of Lemma 1, while implementing a two-sided error algorithm for the Direct Product Problem (of Π) by using the two-sided error ϵ-tester for Π. If the reduction outputs 1, then A outputs 1. Otherwise, let I be the "too big" set considered at the iteration in which the reduction halts with output 0.
2. Algorithm A selects uniformly at random $i \in I$, invokes the one-sided error ϵ-tester for Π on x_i, and outputs the output it has obtained.

By Lemma 1, algorithm A has query complexity $O(m) \cdot \mathtt{Q}_\epsilon(\Pi) + \mathtt{Q}^{\text{ose}}_\epsilon(\Pi)$, and by its construction it only outputs 0 if the one-sided error tester of Π outputs 0 on one of the x_i's. Thus, A has one-sided error. On the other hand, if some of the x_i's are in $\overline{\Gamma}_\epsilon(\Pi)$, then with probability at least $2/3$, algorithm A proceeds to its second step. A closer look at the proof of Lemma 1 reveals that we can guarantee that, in this case, with probability at least $3/4$, at least half of the instances in I are in $\overline{\Gamma}_\epsilon(\Pi)$. (All that is needed is to reduce the error probability on individual instances from $2^{-(j+3)}$ to $2^{-(j+4)}$.) This is the case because, for $j \geq 2$, with probability at least $3/4$, at the beginning of the j^{th} iteration the set I may contain at most $0.5 \cdot 2^{-(j-1)} \cdot m$ instances in Π, whereas halting occurs when $|I| > 2^{-(j-1)} \cdot m$. Hence, with probability at least $\frac{3}{4} \cdot 0.5 \cdot \frac{2}{3}$, algorithm A outputs 0. ◀

6.3 Nonadaptive Algorithms with One-sided Error

The proof of Theorem 7 can be adapted to the case of (nonadaptive) algorithms of one-sided error. All that is required is to be careful about the lower bound arguments. Specifically, when truncating executions that make too many queries to a certain instance, we let the algorithm output 1 (rather than halt with arbitrary output). Actually, we can close the gap left in Theorem 7 (and get a tight result also for the Direct Product Problem) by using the simple reduction of the Direct Product Problem to the Direct Sum Problem mentioned in Section 6.2. Hence, letting $\mathtt{Q}^{\text{dr}}_\epsilon$ denote the minimum query complexity of all nonadaptive one-sided error ϵ-testers for Π, where "dr" stands for doubly restricted, we get:

▶ **Theorem 9** (Doubly Restricted Query Complexity of Multiple-instance Problems)**.** *For every property Π, proximity parameter ϵ, and integer m, the following holds.*

The Direct Sum Problem: *The nonadaptive one-sided error query complexity of $\mathtt{DS}^m_\epsilon(\Pi)$ is $\Theta(m \cdot \mathtt{Q}^{\text{dr}}_\epsilon(\Pi))$.*

The Direct Product Problem: *The nonadaptive one-sided error query complexity of $\mathtt{DP}^m_\epsilon(\Pi)$ is $\Theta(m \cdot \mathtt{Q}^{\text{dr}}_\epsilon(\Pi))$.*

The Concatenation Problem: *For $c \geq 1$, suppose that $\mathtt{Q}_\epsilon(\Pi)$ increases at least linearly with $1/\epsilon^c$. Then, the nonadaptive one-sided error query complexity of $\mathtt{CP}^m_\epsilon(\Pi)$ is $O(\log(1/\epsilon))^2 \cdot \mathtt{Q}^{\mathrm{dr}}_{\epsilon/4}(\Pi)$ if $c = 1$, and $O(\mathtt{Q}^{\mathrm{dr}}_{\epsilon/4}(\Pi))$ otherwise* (i. e., for $c > 1$).

Acknowledgements. We are grateful to Tom Gur and Ron Rothblum for comments on an early draft of this paper. We thank Ilan Newman for calling our attention to the fact that Lemma 1 is implicit in the work of Feige *et al.* [1].

References

1 U. Feige, P. Raghavan, D. Peleg, and E. Upfal. Computing with Noisy Information. *SIAM Journal on Computing*, Vol. 23 (5), pages 1001–1018, 1994.

2 O. Goldreich. *Foundations of Cryptography – Basic Tools*. Cambridge University Press, 2001.

3 O. Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

4 O. Goldreich, editor. *Property Testing – Current Research and Surveys*. Lecture Notes in Computer Science, Vol. 6390, Springer, 2010.

5 O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, pages 653–750, July 1998. Extended abstract in *37th FOCS*, 1996.

6 O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In the proceedings of *21st ACM Symposium on the Theory of Computing*, pages 25–32, 1989.

7 O. Goldreich, N. Nisan and A. Wigderson. On Yao's XOR-Lemma. *ECCC*, TR95-050, 1995.

8 O. Goldreich and D. Ron. Property testing in bounded degree graphs. *Algorithmica*, pages 302–343, 2002. Extended abstract in *29th STOC*, 1997.

9 L. A. Levin. One-way functions and pseudorandom generators. In proc. of the *17th ACM Symposium on the Theory of Computing*, pages 363–365, 1985.

10 I. Newman. Computing in Fault Tolerant Broadcast Networks and Noisy Decision Trees. *Random Struct. Algorithms*, Vol. 34 (4), pages 478–501, 2009.

11 D. Ron. Property testing: A learning theory perspective. *Foundations and Trends in Machine Learning*, Vol. 1 (3), pages 307–402, 2008.

12 D. Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends in TCS*, Vol. 5 (2), pages 73–205, 2009.

13 R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2), pages 252–271, 1996.

14 J. von Neumann. Various techniques used in connection with random digits. *Applied Math Series*, Vol. 12, pages 36–38, 1951. Reprinted in *von Neumann's Collected Works*, Vol. 5, pages 768–770, Pergamon, 1963.

15 A. C. C. Yao. Lower Bounds by Probabilistic Arguments (Extended Abstract). In *Proc. 24th IEEE Symposium on Foundations of Computer Science*, pages 420–428, 1983.

A In Passing

In this appendix, we elaborate on two comments that were made in the main text. Section A.1 discusses a well-known result, while expressing an opinion of its contents. Section A.2 explicitly presents a useful idea, which is implicit in many prior works (as well as in Section 5).

A.1 On Yao's MiniMax Principle

As is well known, Yao [15] made the influential observation that von Neumann's MiniMax Principle for zero-sum games [14] can be applied in the context of the analysis of randomized algorithms. In particular, Yao considered an "algorithmic" player selecting a distribution over deterministic algorithms and an "adversarial (input)" player selecting an input distribution. Then, the application of the game theoretic principle implies that the profit (e. g., success probability) of the best randomized algorithm on the worst-case input equals the upper bound on the profit of any (deterministic) algorithm under some fixed distribution of inputs. (The same applies when one consider the cost (e. g., in resources) incurred by the algorithm.)

Our point is that the above statement has two directions. The first direction (which is obvious and more commonly used) is that the profit of the best algorithm on the worst-case input cannot exceed the profit any algorithm may get under some (adversarially chosen) input distribution. The second direction, which is the remarkable one, asserts that the upper bound obtained in this way (i. e., by a worst adversarial selection of an input distribution) is tight. In other words, the foregoing method of bounding the profit of randomized algorithms is complete (or optimal). Thus, in our opinion, Yao contribution was in pioneering and advocating the method of obtaining lower bounds via the design of adversarial input distributions, not in proving the soundness of this method. (Again, the method's soundness is obvious, what is remarkable is its completeness.)

For sake of clarity, let us cast the above discussion in precise terms. Let $p(r,c)$ be the profit attained by the row player, when it picks the row r and the column player picks c. Let V be the value obtained when the row player picks r under the best possible distribution on rows, denoted $\mathcal{R}$, and the column player picks c to minimize the row player's profit; that is, $V \stackrel{\text{def}}{=} \min_c\{\mathrm{E}_{r\leftarrow\mathcal{R}}[p(r,c)]\}$. Let U be the upper bound on the profit obtained when the column player uses a distribution, denoted $\mathcal{C}$, that minimizes the profit of the best choice of a row r; that is, $U \stackrel{\text{def}}{=} \max_r\{\mathrm{E}_{c\leftarrow\mathcal{C}}[p(r,c)]\}$. Obviously, $V \leq U$:

$$\begin{aligned} V &= \min_c\{\mathrm{E}_{r\leftarrow\mathcal{R}}[p(r,c)]\} \\ &\leq \mathrm{E}_{r\leftarrow\mathcal{R},c\leftarrow\mathcal{C}}[p(r,c)] \\ &\leq \max_r\{\mathrm{E}_{c\leftarrow\mathcal{C}}[p(r,c)]\} \\ &= U. \end{aligned}$$

The actual contents of the MiniMax Principle is that this upper bound is actually tight; that is, $V = U$. The known proofs of the latter assertion are far more complex than the above manipulation: The classical proof proceeds by presenting a transformation over the space of strategy pairs such that its fixed-points are equilibria pairs, and applying Brouwer's fixed-point theorem. A popular alternative proof proceeds by formulating the problem of finding an optimal strategy (for the row player) as a linear program and applying the strong LP-Duality Theorem.

Summary. We claim that three things are being confused: (1) The generic fact that $V \leq U$; (2) the generic fact that V actually equals U; and (3) the suggestion that in many specific

settings (e. g., where the payoff represents the success probability of an algorithm on an input) it is beneficial to prove upper bounds on V by proving upper bounds on U.

A.2 On Levin's Economical Work Investment Strategy

In some situations one can sample a huge space that contains elements of different quality such that elements of lower quality require more work to utilize. The aim is to utilize some element, but the work required for utilizing the various elements is not known a priori, and it only becomes known after the entire amount of required work is invested. Note that it may be that most of the elements are of very poor quality, and so it is not a good idea to select a single element and invest as much work as is needed to utilize it. Instead one may want to select many samples points and invest in each of them a limited amount of work (which may be viewed as probing the required amount of work).

To be more concrete, suppose that the work that needs to be invested in a sample point s (in order to utilize it) is inversely proportional to its (unknown to us) quality $q(s)$. We only know a lower bound ϵ on the average quality of an element (i. e., $\mathrm{E}_s[q(s)] > \epsilon$), and we wish to minimize the total amount of work invested in utilizing some element. One natural strategy that comes to mind is to sample $O(1/\epsilon)$ points and invest $O(1/\epsilon)$ work in each of these points. In this case we succeed with constant probability, while investing $O(1/\epsilon^2)$ work. The analysis is based on the fact that $\mathrm{E}_s[q(s)] > \epsilon$ implies that $\Pr_s[q(s) > \epsilon/2] > \epsilon/2$. The following fact suggests a more economical strategy.

▶ **Fact A.1** (A Refined Counting Argument). *Let $\mathcal{D}$ be a probability distribution, $q : \mathrm{Supp}(\mathcal{D}) \to [0,1]$, and $\epsilon \in (0,1]$. Suppose that $\mathrm{E}_{s\leftarrow\mathcal{D}}[q(s)] > \epsilon$, and let $\ell = \lceil \log_2(2/\epsilon) \rceil$. Then, there exists $j \in [\ell]$ such that $\Pr_{s\leftarrow\mathcal{D}}[q(s) > 2^{-j}] > 2^j\epsilon/4\ell$.*

Hence, an alternative strategy can succeed, with constant probability, by investing $\widetilde{O}(1/\epsilon)$ work. Specifically, for each $j \in [\ell]$, we take $O(\ell/2^j\epsilon)$ samples, and invest $O(2^j)$ work in each of these sample points. (Note that we cannot expect to invest less than $o(1/\epsilon)$ work in total, since we may have $q(s) = 2\epsilon$ for each sample point s, and so the alternative strategy is almost optimal.)

We learned this alternative strategy from Leonid Levin in the mid-1980s. This strategy is used in [9] (see the last paragraph of [9, Sec. 9]) and is stated explicitly in [6, Lem. 3] (see [2, Clm. 2.5.4.1] for an alternative presentation). Within the context of property testing, this strategy was first used in [8] (see Lemma 3.3 in the proceeding version and Lemma 3.6 in the journal version).

Proof. Let $B_j \stackrel{\text{def}}{=} \{s : 2^{-j} < q(s) \leq 2^{-(j-1)}\}$, and assume towards the contradiction that for every $j \in [\ell]$ it holds that $\Pr_s[q(s) > 2^{-j}] \leq 2^j\epsilon/4\ell$. This implies that for every $j \in [\ell]$ it holds that $\Pr_s[s \in B_j] \leq 2^j\epsilon/4\ell$ (and $q(s) \leq \epsilon/2$ for every $s \notin \bigcup_{j\in[\ell]} B_j$). We get

$$\begin{aligned}
\mathrm{E}_s[q(s)] &\leq \frac{\epsilon}{2} + \sum_{j\in[\ell]} \Pr_s[s \in B_j] \cdot 2^{-(j-1)} \\
&\leq \frac{\epsilon}{2} + \sum_{j\in[\ell]} \frac{2^j\epsilon}{4\ell} \cdot 2^{-(j-1)} \\
&= \frac{\epsilon}{2} + \sum_{j\in[\ell]} \frac{\epsilon}{2\ell}
\end{aligned}$$

which contradicts the fact's hypothesis. ◀

The Case of Required Work that Increases Faster than $O(1/q(\cdot))$. The above description refers to the case that the work that needs to be invested in utilizing an element is inversely proportional to its quality (i. e., the work that needs to be invested in s is $\Theta(1/q(s))$). In that case, we sought a set S such that the product $\Pr_s[s \in S] \cdot \min_{s \in S}\{q(s)\}$ is maximized. (Actually, we identified $O(\log(1/\epsilon))$ candidate sets S, and invested $\widetilde{O}(1/\epsilon)$ work in each of them.) We now consider the case that for some $c > 1$ (e. g., $c = 2$), the work that needs to be invested in order to utilize the element s is $\Theta(1/q(s)^c)$. In this case, we seek a set S such that the product $\Pr_s[s \in S] \cdot \min_{s \in S}\{q(s)^c\}$ is maximized. (Indeed, the case of $c = 2$ arises quite often in applications; for example, it arises whenever one wishes to approximate some $[0,1]$-valued quantity up to $\pm q(s)$.)

Here the straightforward solution is to sample $O(1/\epsilon)$ points and invest $O(1/\epsilon^c)$ work in each of these sample points. In this case we succeed with constant probability, while investing $O(1/\epsilon^{c+1})$ work. The following fact suggests a more economical procedure.

▶ **Fact A.2** (An Alternatively Refined Counting Argument). *Let $\mathcal{D}$ be a probability distribution, $q : \mathrm{Supp}(\mathcal{D}) \to [0,1]$, and $\epsilon \in (0,1]$. Suppose that $\mathrm{E}_{s \leftarrow \mathcal{D}}[q(s)] > \epsilon$, and let $\ell = \lceil \log_2(2/\epsilon) \rceil$. Then, there exists $j \in [\ell]$ such that $\Pr_{s \leftarrow \mathcal{D}}[q(s) > 2^{-j}] > 2^j\epsilon/(\ell + 5 - j)^2$.*

Hence, an alternative strategy can succeed with constant probability by investing $O(1/\epsilon^c)$ work, which is optimal (since we may have $q(s) = 2\epsilon$ for all s). Specifically, for each $j \in [\ell]$, we take $O((\ell + 5 - j)^2/2^j\epsilon)$ samples, and invest $O(2^{cj})$ work in each sample point. Indeed, for every $c > 1$, it holds that

$$\begin{aligned}\sum_{j \in [\ell]} \frac{(\ell + 5 - j)^2}{2^j\epsilon} \cdot 2^{cj} &= \frac{1}{\epsilon} \cdot \sum_{k \in [\ell]} (k+4)^2 \cdot 2^{(c-1)\cdot(\ell+1-k)} \\ &= O(1/\epsilon)^c \cdot \sum_{k \in [\ell]} (k+4)^2 \cdot 2^{-(c-1)\cdot k}\end{aligned}$$

which equals $O(1/\epsilon^c)$, because $\sum_{k \in [\ell]} (k+4)^2 \cdot 2^{-c'k} = O(1)$ for every $c' > 0$.

Proof. The proof is by a simple adaptation of the proof of Fact A.1. As before, let $B_j \stackrel{\text{def}}{=} \{s : 2^{-j} < q(s) \leq 2^{-(j-1)}\}$, and assume towards the contradiction that for every $j \in [\ell]$ it holds that $\Pr_s[q(s) > 2^{-j}] \leq 2^j\epsilon/(\ell + 5 - j)^2$. This implies that for every $j \in [\ell]$ it holds that $\Pr_s[s \in B_j] \leq 2^j\epsilon/(\ell + 5 - j)^2$ (and $q(s) \leq \epsilon/2$ for every $s \notin \bigcup_{j \in [\ell]} B_j$). We get

$$\begin{aligned}\mathrm{E}_s[q(s)] &\leq \frac{\epsilon}{2} + \sum_{j \in [\ell]} \Pr_s[s \in B_j] \cdot 2^{-(j-1)} \\ &\leq \frac{\epsilon}{2} + \sum_{j \in [\ell]} \frac{2^j\epsilon}{(\ell + 5 - j)^2} \cdot 2^{-(j-1)} \\ &= \frac{\epsilon}{2} + \sum_{k \in [\ell]} \frac{2\epsilon}{(k+4)^2}\end{aligned}$$

which contradicts the fact's hypothesis (since $\sum_{k \in [\ell]} (k+4)^{-2} < 1/4$). ◀

Communication Complexity of Set-Disjointness for All Probabilities*

Mika Göös and Thomas Watson

Department of Computer Science, University of Toronto
Toronto, ON, Canada
{mgoos,thomasw}@cs.toronto.edu

Abstract

We study set-disjointness in a generalized model of randomized two-party communication where the probability of acceptance must be at least $\alpha(n)$ on yes-inputs and at most $\beta(n)$ on no-inputs, for some functions $\alpha(n) > \beta(n)$. Our main result is a complete characterization of the private-coin communication complexity of set-disjointness for all functions α and β, and a near-complete characterization for public-coin protocols. In particular, we obtain a simple proof of a theorem of Braverman and Moitra (STOC 2013), who studied the case where $\alpha = 1/2 + \epsilon(n)$ and $\beta = 1/2 - \epsilon(n)$. The following contributions play a crucial role in our characterization and are interesting in their own right.

1. We introduce two communication analogues of the classical complexity class that captures *small bounded-error* computations: we define a "restricted" class SBP (which lies between MA and AM) and an "unrestricted" class USBP. The distinction between them is analogous to the distinction between the well-known communication classes PP and UPP.
2. We show that the SBP communication complexity is precisely captured by the classical *corruption* lower bound method. This sharpens a theorem of Klauck (CCC 2003).
3. We use information complexity arguments to prove a linear lower bound on the USBP complexity of set-disjointness.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases Communication Complexity, Set-Disjointness, All Probabilities

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.721

1 Introduction

In the *set-disjointness* problem, Alice is given an $x \subseteq [n]$, Bob is given a $y \subseteq [n]$, and their task is to decide whether $x \cap y = \emptyset$. Equivalently, viewing x and y as binary strings, we define

$$\textsc{Disj}(x, y) := \neg \bigvee_{i \in [n]} (x_i \wedge y_i) .$$

Set-disjointness is the preeminent coNP-complete problem in communication complexity [2, 13]. A fundamental result of Kalyanasundaram and Schnitger [23] (with alternative proofs given by [31, 4]) states that every randomized protocol for set-disjointness requires $\Omega(n)$ bits of communication to achieve a constant error probability that is bounded away from 1/2. These lower bounds have been extremely useful in applications of communication complexity to other areas of theoretical computer science, including circuit complexity,

* This work was supported by funding from NSERC.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 721–736

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

distributed computing, streaming, data structures, combinatorial optimization, and more; see [28, 22, 13].

In this work, we study set-disjointness in a generalized setting where the probability of acceptance must be at least $\alpha(n)$ on yes-inputs and at most $\beta(n)$ on no-inputs, for any prescribed functions $\alpha(n) > \beta(n)$.

1.1 Main Result

Our main result is a complete characterization of the private-coin communication complexity of set-disjointness for all functions α and β, and a near-complete characterization for public-coin protocols. Roughly speaking, we prove that the randomized complexity is

$$\Theta(n \cdot (1 - \beta/\alpha))$$

for typical functions α and β; see subsection 1.4 for the statement of the exact bounds.

As a special case, we obtain a simple proof of a result of Braverman and Moitra [6]. They showed that the communication complexity of set-disjointness is $\Theta(\epsilon n)$ in case $\alpha = 1/2 + \epsilon(n)$ and $\beta = 1/2 - \epsilon(n)$. While this special case might suggest that the complexity is determined by the additive gap $\alpha - \beta$, our characterization reveals that, in fact:

> **Central tenet:** *It is not the additive gap between α and β that determines the complexity of set-disjointness; what matters is the multiplicative gap.*

Our proof follows this ideology: we show that in order to understand the communication complexity for all α and β it suffices to understand the *small bounded-error* case where α is tiny (e.g., exponentially small in n) and $\beta = \alpha/2$.

1.2 SBP: Small Bounded-error Probabilities

In classical time-bounded (i.e., poly-time Turing machine) complexity theory, small bounded-error acceptance probabilities are captured by a counting class called SBP, which was introduced by Böhler, Glaßer, and Meister [5] and has also been studied in [38]. In particular, [5] observed that SBP is sandwiched between the Arthur–Merlin classes MA and AM [3].

In this work, we introduce two communication complexity analogues of SBP: a *restricted* class called SBP, and an *unrestricted* class called USBP. These classes are natural and interesting in their own right. Most importantly, they serve to structure our argument.

Randomized Communication Complexity. In what follows, we assume familiarity with basic definitions of communication complexity [28, 22]. Fix a two-party function $f\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ where on input (x, y) Alice is given x and Bob is given y. We say (x, y) is a b-input if $(x, y) \in f^{-1}(b)$. We let $\mathsf{R}^{\mathsf{pub}}_{\alpha,\beta}(f)$, respectively $\mathsf{R}^{\mathsf{priv}}_{\alpha,\beta}(f)$, denote the minimum communication complexity (as a function of n) of a public-coin, respectively private-coin, protocol for f such that the probability of acceptance is at least $\alpha(n)$ on all 1-inputs and at most $\beta(n)$ on all 0-inputs. As is customary [2], for any communication measure $\mathsf{C}(f)$ we often let C stand for the class of functions f with $\mathsf{C}(f) = \mathrm{polylog}(n)$.

PP and UPP. To motivate our upcoming definitions for SBP, we take a little detour and recall the communication classes associated with the standard complexity class PP. There

are in fact two distinct measures—*restricted* and *unrestricted*—as introduced in [2, 30]:

$$\mathsf{PP}(f) \coloneqq \min_{\epsilon(n)>0} \mathsf{R}^{\mathsf{pub}}_{1/2+\epsilon,\,1/2-\epsilon}(f) + \log(1/\epsilon),$$
$$\mathsf{UPP}(f) \coloneqq \min_{\epsilon(n)>0} \mathsf{R}^{\mathsf{priv}}_{1/2+\epsilon,\,1/2-\epsilon}(f).$$

In the (restricted) public-coin model, one needs to charge the additional $\log(1/\epsilon)$ term in order for the measure to be well-behaved when ϵ is tiny. (For example, note that $\mathsf{R}^{\mathsf{pub}}_{1/2+\epsilon,\,1/2-\epsilon}(f) \leq 2$ for $\epsilon = 2^{-n-1}$.) The original definition of $\mathsf{PP}(f)$ given in [2] actually charged for the number of public coin flips *instead* of the $+\log(1/\epsilon)$; however, by standard sparsification techniques (see [29] and [28, Theorem 3.14]) the two versions are essentially equivalent—they are within a constant factor plus $O(\log n)$—and the definition we have stated is much more prevalent in recent literature. It also follows from standard sparsification that we may convert any PP protocol into a UPP protocol of comparable cost: $\mathsf{UPP}(f) \leq O(\mathsf{PP}(f) + \log n)$. In the converse direction, an exponential separation between UPP and PP is known [8, 33, 34].

SBP and USBP. Analogously to the above, we define

$$\mathsf{SBP}(f) \coloneqq \min_{\alpha(n)>0} \mathsf{R}^{\mathsf{pub}}_{\alpha,\,\alpha/2}(f) + \log(1/\alpha),$$
$$\mathsf{USBP}(f) \coloneqq \min_{\alpha(n)>0} \mathsf{R}^{\mathsf{priv}}_{\alpha,\,\alpha/2}(f).$$

Here the constant factor $1/2 = \beta/\alpha$ can be replaced by any positive constant less than 1 while affecting the complexity measures by only a constant factor: if we run a protocol ℓ times and accept iff all iterations accept, then β/α gets raised to the power ℓ while the communication and the $\log(1/\alpha)$ term each get multiplied by ℓ. We call this procedure *and-amplification* (in contrast to the usual *majority-amplification*). We also note that by standard sparsification, $\mathsf{USBP}(f) \leq O(\mathsf{SBP}(f) + \log n)$ holds for all f. In the converse direction, we do not know whether USBP is significantly more powerful than SBP (though a small separation is witnessed by the greater-than function, which has constant USBP complexity but $\Theta(\log n)$ SBP and PP complexity [7]).

Relationship to Arthur–Merlin Classes. Klauck [25, 27] and Aaronson and Wigderson [1] took up the study of communication complexity analogues of Arthur–Merlin games. Their results have already found applications in data streaming [9, 10, 19]. We do not define the communication models MA and AM here, but we note that the classical inclusions continue to hold in the communication setting (for the same reasons):

$$\mathsf{MA} \subseteq \mathsf{SBP} \subseteq \mathsf{AM}.$$

Indeed, if $\mathsf{MA}(f) = m$ then by majority-amplification and by absorbing Merlin's nondeterminism into the randomness we obtain $\mathsf{R}^{\mathsf{pub}}_{2^{-m-1},\,2^{-m-2}}(f) \leq O(m^2)$. Thus $\mathsf{SBP}(f) \leq O(\mathsf{MA}(f)^2)$ (and the quadratic blow-up is necessary for "black-box" simulations [15]). On the other hand, $\mathsf{AM}(f) \leq O(\mathsf{SBP}(f) + \log n)$ holds by sparsifying the randomness and using the Goldwasser–Sipser protocol [18].

1.3 Results for SBP and USBP

We prove that SBP communication complexity is exactly characterized by the well-known *corruption* lower bound method (also known as the *rectangle bound* or *one-sided discrepancy*).

The definition of the corruption bound $\mathsf{Corr}(f)$ is given in section 2, but for now, we note that $\mathsf{Corr}(f)$ essentially depends on the size of the largest approximately 1-monochromatic rectangle in the communication matrix of f. (For an extensive discussion of the different lower bound methods in communication complexity, see [20].) Previously, Klauck [25] showed that $\mathsf{Corr}(f)$ lies somewhere between the MA and AM communication complexities of f; namely $\Omega(\mathsf{AM}(f)) \leq \mathsf{Corr}(f) \leq O(\mathsf{MA}(f)^2)$. Klauck also gave a combinatorial near-characterization of $\mathsf{Corr}(f)$ (tight up to logarithmic factors) using so-called *one-sided uniform threshold covers*. The following theorem sharpens these results by pinpointing precisely the class between MA and AM that is characterized by corruption.

▶ **Theorem 1.** $\mathsf{SBP}(f) = \Theta(\mathsf{Corr}(f))$ *for all* f.

One way to frame Theorem 1 is as follows. A lot of effort (e.g., [25, 20, 24, 21, 17]) has been spent on comparing the relative strengths of different lower bound methods in communication complexity with the goal of finding a natural method that captures the bounded-error randomized communication complexity of every function. Theorem 1 can be viewed as achieving a diametrically opposite goal: we start with a historically important lower bound method (i.e., corruption) and find a natural communication measure that it captures. Theorem 1 is also somewhat analogous, in content and proof, to another result of Klauck [26] showing that the discrepancy bound captures PP.

Razborov [31] famously proved that $\mathsf{Corr}(\textsc{Disj}) = \Theta(n)$. (The first linear lower bound for set-disjointness [23] did not use corruption.) By the results of [25], this implies that $\mathsf{MA}(\textsc{Disj}) \geq \Omega(\sqrt{n})$. We immediately have a stronger corollary.

▶ **Corollary 2.** $\mathsf{SBP}(\textsc{Disj}) = \Theta(n)$.

While the classical rectangle-based methods suffice to analyze SBP protocols, these techniques are not well-suited for handling acceptance probabilities $\alpha(n)$ that are arbitrarily small functions of n (e.g., doubly exponentially small in n). To obtain lower bounds for USBP we pursue a different avenue and show that the *information complexity* framework, as formulated by Bar-Yossef, Jayram, Kumar, and Sivakumar [4] (see also [12]), can be adapted to suit our purposes. The main technical result of this work is the following, proved in section 3.

▶ **Theorem 3.** $\mathsf{USBP}(\textsc{Disj}) = \Theta(n)$.

We note that the statement of Theorem 3 is similar in spirit to Forster's theorem [16] stating that the UPP complexity of the inner product function is $\Theta(n)$. Note also that Theorem 2 is of course a corollary of Theorem 3, too, but the corruption-based proof via Theorem 1 is arguably more elementary than the proof of Theorem 3. Finally, we note that the well-studied gap-Hamming-distance promise problem [11, 37, 36] (where 1-inputs have distance $\geq \frac{n}{2} + \sqrt{n}$ and 0-inputs have distance $\leq \frac{n}{2} - \sqrt{n}$) has SBP and USBP complexities $\Theta(\sqrt{n})$, where the lower bound follows by Theorem 3 and a standard reduction from Disj, and the upper bound follows by and-amplification of the trivial protocol that checks inequality at a random bit position.

1.4 Characterization for All α and β

Using our results for SBP and USBP in a black-box manner we derive the following (near) complete characterization for the randomized communication complexity of set-disjointness in section 4.

▶ **Theorem 4** (Private-coin). *For all $\alpha(n) > \beta(n)$,*

$$\mathsf{R}^{\mathsf{priv}}_{\alpha,\beta}(\textsc{Disj}) = \Theta(n \cdot (1-\beta/\alpha) + \log n).$$

▶ **Theorem 5** (Public-coin). *There is a universal constant $C > 0$ such that for all $\alpha(n) > \beta(n)$,*

$$\mathsf{R}^{\mathsf{pub}}_{\alpha,\beta}(\textsc{Disj}) = \begin{cases} \Theta(n \cdot (1-\beta/\alpha)) & \text{when} \quad \log(1/\alpha) \le C \cdot n \cdot (1-\beta/\alpha), \\ 2 & \text{when} \quad \log(1/\alpha) \ge \lceil n \cdot (1-\beta/\alpha) \rceil. \end{cases}$$

We stress that for the public-coin characterization (and in particular, the result of [6] as a corollary), it suffices to rely only on Razborov's corruption lemma (via Theorem 2), and not on any information complexity techniques. Braverman and Moitra [6] observed that $\mathsf{R}^{\mathsf{pub}}_{1/2+\epsilon,\,1/2-\epsilon}(\textsc{Disj}) \ge \Omega(\epsilon^2 n)$ follows from the standard bounded-error lower bound by majority-amplification, and they obtained the tight $\Omega(\epsilon n)$ bound by developing information complexity techniques tailored to this setting. Our idea is that and-amplification imposes only an ϵ factor loss (rather than the ϵ^2 factor loss imposed by majority-amplification) while still reducing to a case where the corruption method applies.

We also note that for public-coin protocols there remains a small gap in the parameters around the threshold $\log(1/\alpha) = \Theta(n \cdot (1-\beta/\alpha))$ that is not covered by our theorem. As we discuss in section 4, the power of the public coins kicks in at this threshold.

Finally, we mention that all the set-disjointness lower bounds in this paper continue to hold under the *unique-intersection* promise where the inputs are either disjoint or intersect in exactly one coordinate: for Theorem 2 this property is inherited from Razborov's proof; for Theorem 3 this property is implicit in our proof.

2 SBP is Characterized by Corruption

In this section we prove Theorem 1, which states that $\mathsf{SBP}(f) = \Theta(\mathsf{Corr}(f))$ for all f. We start by defining the corruption bound. We say a distribution μ over inputs is *balanced* (with respect to f) if $\mu(f^{-1}(1)) = \mu(f^{-1}(0)) = 1/2$. We say a rectangle R is 1-*biased* (with respect to f and μ) if $\mu(R \cap f^{-1}(0)) \le \mu(R)/8$. The corruption bound is defined as

$$\mathsf{Corr}(f) := \max_{\text{balanced } \mu} \min_{\text{1-biased } R} \log\left(\frac{1}{\mu(R)}\right).$$

It was proved in [25] that the constant factor of $1/8$ (in the definition of 1-biased) can be replaced by any positive constant at most $1/8$ while affecting the corruption bound by only a constant factor. It was also proved in [25] that the bound is robust with respect to the balance condition on μ.

2.1 SBP is Lower Bounded by Corruption

Here we show the lower bound $\mathsf{SBP}(f) \ge \Omega(\mathsf{Corr}(f))$. The intuition is as follows. The first step is to fix the public randomness of an SBP protocol in such a way that the average-case behavior of the resulting deterministic protocol mimics the worst-case behavior of the original protocol. Typically, this sort of thing is done by invoking the distributive law (linearity of expectation), but here we need a more elaborate calculation due to the asymmetric nature of SBP. Then, the rest of the argument follows along similar lines as the proof in [25] (that $\mathsf{Corr}(f) \le O(\mathsf{MA}(f)^2)$), showing that the 1-inputs are mostly covered by "small" transcript rectangles (of our average-case protocol), hence many such rectangles are needed.

We proceed with the formal proof. Let Π be an $\mathsf{R}^{\mathsf{pub}}_{\alpha,\alpha/32}$ protocol for f; recall that by and-amplification we may assume $\beta = \alpha/32$ rather than $\beta = \alpha/2$ in the definition of SBP. Assuming $\log(1/\alpha) < \mathsf{Corr}(f)/2$, we show that Π uses $\Omega(\mathsf{Corr}(f))$ bits of communication. To this end, fix a balanced distribution μ such that for all 1-biased rectangles R, $\mu(R) \le 2^{-\mathsf{Corr}(f)}$.

Identify the possible outcomes of public randomness with $\{1,\dots,m\}$, and let Π_i denote Π running with public randomness i. Let p_i be the probability the public randomness is i (so $p_i = 1/m$ if the public randomness is uniformly distributed). Let q_i be the probability over μ that Π_i accepts, conditioned on the input being a 1-input. Let r_i be the same but conditioned on a 0-input. Now

$$\sum_i p_i q_i = \Pr_{i,(x,y)\sim\mu}\big[\,\Pi_i(x,y) \text{ accepts} \;\big|\; f(x,y)=1\,\big] \ge \alpha, \tag{1}$$

$$\sum_i p_i r_i = \Pr_{i,(x,y)\sim\mu}\big[\,\Pi_i(x,y) \text{ accepts} \;\big|\; f(x,y)=0\,\big] \le \alpha/32. \tag{2}$$

▶ Claim 6. There exists an i^* such that $q_{i^*} \ge \alpha/2$ and $r_{i^*} \le q_{i^*}/16$.

Proof of claim. Suppose for contradiction that for all i either $q_i < \alpha/2$ or $r_i > q_i/16$. Let $S \subseteq \{1,\dots,m\}$ be such that for all $i \in S$, $q_i < \alpha/2$, and for all $i \in \overline{S}$, $r_i > q_i/16$. Then

$$\begin{aligned}\sum_i p_i r_i &\ge \sum_{i\in\overline{S}} p_i r_i \ge \sum_{i\in\overline{S}} p_i q_i/16 \\ &= \frac{1}{16}\Big(\sum_i p_i q_i - \sum_{i\in S} p_i q_i\Big) \ge \frac{1}{16}\Big(\alpha - \Big(\sum_{i\in S} p_i\Big)\alpha/2\Big) \ge \alpha/32.\end{aligned}$$

Furthermore, at least one of the inequalities must be strict, contradicting (2). ◀

Fix an i^* guaranteed by Claim 6. Using i^* as the public randomness in Π, we can now apply the usual corruption argument. Consider the 1-rectangles that correspond to accepting transcripts of Π_{i^*}. Call a 1-rectangle R *large* if $\mu(R) > 2^{-\mathsf{Corr}(f)}$ and *small* otherwise. Recall that by our assumption on μ, no large 1-rectangle is 1-biased: for every large 1-rectangle R we have $\mu(R \cap f^{-1}(0)) > \mu(R)/8$. Under μ, the total measure of large 1-rectangles is at most half the total measure of all 1-rectangles, since otherwise

$$\begin{aligned} r_{i^*} &= 2\Pr_{(x,y)\sim\mu}\big[\,\Pi_{i^*}(x,y) \text{ accepts and } f(x,y)=0\,\big] \\ &= 2\textstyle\sum_{\text{1-rectangles } R} \mu(R \cap f^{-1}(0)) \\ &\ge 2\textstyle\sum_{\text{large 1-rectangles } R} \mu(R \cap f^{-1}(0)) \\ &> \tfrac{1}{4}\textstyle\sum_{\text{large 1-rectangles } R} \mu(R) \\ &> \tfrac{1}{4}\cdot\tfrac{1}{2}\textstyle\sum_{\text{1-rectangles } R} \mu(R) \\ &\ge \tfrac{1}{8}\textstyle\sum_{\text{1-rectangles } R} \mu(R \cap f^{-1}(1)) \\ &= \tfrac{1}{8}\Pr_{(x,y)\sim\mu}\big[\,\Pi_{i^*}(x,y) \text{ accepts and } f(x,y)=1\,\big] \\ &= \tfrac{1}{8}\cdot q_{i^*}/2 \\ &= q_{i^*}/16. \end{aligned}$$

Therefore $\sum_{\text{small 1-rectangles } R} \mu(R) \ge \frac{1}{2}\sum_{\text{1-rectangles } R} \mu(R) \ge \frac{1}{2}\cdot q_{i^*}/2 \ge \alpha/8 > 2^{-\mathsf{Corr}(f)/2-3}$. Thus there are at least $2^{-\mathsf{Corr}(f)/2-3}/2^{-\mathsf{Corr}(f)} = 2^{\mathsf{Corr}(f)/2-3}$ small 1-rectangles, which implies that Π uses at least $\mathsf{Corr}(f)/2 - 3$ bits of communication.

2.2 SBP is Upper Bounded by Corruption

Here we show the upper bound $\mathsf{SBP}(f) \leq O(\mathsf{Corr}(f))$. The intuition is as follows. If the corruption bound is small, that means for every balanced distribution over inputs there exists a rectangle (which can be viewed as a 2-bit protocol) that exhibits average-case SBP-like behavior—accepting a random 1-input with not-too-small probability, and accepting a random 0-input with constant-factor-smaller probability. We use the minimax theorem to convert this property into a distribution over rectangles, with a worst-case SBP guarantee. Several technical issues arise with this argument. One is the asymmetry between 1-inputs and 0-inputs, but this can be massaged away using a linear transformation of probabilities before invoking minimax. Another is that the corruption bound can yield an average-case SBP rectangle with a different "α" for different balanced distributions, whereas the minimax application requires a single α to work uniformly for all balanced distributions. This issue is fixed by passing to an appropriate subrectangle to decrease the α if necessary, for any given balanced distribution.

We proceed with the formal proof. For notational convenience we let $\mathbf{0}$ and $\mathbf{1}$ stand for the events $f^{-1}(0)$ and $f^{-1}(1)$, respectively. For example, $\mu(\mathbf{0} \,|\, R) = \mu(R \cap f^{-1}(0))/\mu(R)$ and $\mu(R \,|\, \mathbf{0}) = \mu(R \cap f^{-1}(0))/\mu(f^{-1}(0))$.

Define $\alpha = 2^{-\mathsf{Corr}(f)}$.

▶ Claim 7. For every balanced μ there exists a rectangle R with $\mu(R \,|\, \mathbf{1}) \geq \alpha$ and $\mu(R \,|\, \mathbf{0}) \leq \alpha/2$.

Proof of claim. Fix a balanced distribution μ. By definition of corruption, there exists a rectangle S such that $\mu(S) \geq \alpha$ and $\mu(\mathbf{0} \,|\, S) \leq 1/8$. Decompose S as the disjoint union $S_1 \cup S_2 \cup \cdots \cup S_m$ where the S_i's are the individual rows of S, sorted in nondecreasing order of $\mu(\mathbf{0} \,|\, S_i)$. Let $R_i = S_1 \cup S_2 \cup \cdots \cup S_i$. For every i we know that $\mu(\mathbf{0} \,|\, R_i) \leq \mu(\mathbf{0} \,|\, S) \leq 1/8$. If there exists an i such that $\alpha \leq \mu(R_i \,|\, \mathbf{1}) \leq 2\alpha$ then $R = R_i$ witnesses the claim since

$$\mu(R_i \,|\, \mathbf{0}) = \frac{\mu(\mathbf{0} \,|\, R_i) \cdot \mu(R_i \,|\, \mathbf{1}) \cdot \mu(\mathbf{1})}{\mu(\mathbf{0}) \cdot \mu(\mathbf{1} \,|\, R_i)} \leq \frac{(1/8) \cdot 2\alpha \cdot (1/2)}{(1/2) \cdot (7/8)} = 2\alpha/7 \leq \alpha/2.$$

Otherwise, since $\mu(R_m \,|\, \mathbf{1}) = \mu(S \,|\, \mathbf{1}) = \mu(\mathbf{1} \,|\, S) \cdot \mu(S)/\mu(\mathbf{1}) \geq (7/8) \cdot \alpha/(1/2) > \alpha$ and $\mu(R_0 \,|\, \mathbf{1}) = 0 < \alpha$, there must exist an i such that $\mu(R_i \,|\, \mathbf{1}) > 2\alpha$ and $\mu(R_{i-1} \,|\, \mathbf{1}) < \alpha$ and thus $\mu(S_i \,|\, \mathbf{1}) > 2\alpha - \alpha = \alpha$. In this case, the rectangle $R = S_i \cap \mathbf{1}$ witnesses the claim since $\mu(S_i \cap \mathbf{1} \,|\, \mathbf{1}) = \mu(S_i \,|\, \mathbf{1}) > \alpha$ and $\mu(S_i \cap \mathbf{1} \,|\, \mathbf{0}) = 0 \leq \alpha/2$. ◀

Let M be the matrix with rows indexed by inputs $(x, y) \in \{0,1\}^n \times \{0,1\}^n$ and columns indexed by rectangles $R \subseteq \{0,1\}^n \times \{0,1\}^n$ such that

$$M_{(x,y),R} = \begin{cases} 1 & \text{if } f(x,y) = 1 \text{ and } (x,y) \in R \\ 0 & \text{if } f(x,y) = 1 \text{ and } (x,y) \notin R \\ 0 & \text{if } f(x,y) = 0 \text{ and } (x,y) \in R \\ \frac{\alpha}{1-\alpha/2} & \text{if } f(x,y) = 0 \text{ and } (x,y) \notin R \end{cases}.$$

We claim that for every distribution μ over inputs, there exists a rectangle R such that $\mathbb{E}(M_{\mu,R}) \geq \alpha$ (where $\mathbb{E}$ denotes expectation). If $\mu(\mathbf{0}) = 0$ then take $R = \{0,1\}^n \times \{0,1\}^n$, and if $\mu(\mathbf{1}) = 0$ then take $R = \emptyset$. Otherwise, let μ' be the balanced version of μ and invoke Claim 7 to find an R such that $\mu'(R \,|\, \mathbf{1}) \geq \alpha$ and $\mu'(R \,|\, \mathbf{0}) \leq \alpha/2$. Then we have

$$\mathbb{E}(M_{\mu,R}) \;=\; \mu(R \,|\, \mathbf{1}) \cdot \mu(\mathbf{1}) + \tfrac{\alpha}{1-\alpha/2} \cdot \mu(\overline{R} \,|\, \mathbf{0}) \cdot \mu(\mathbf{0})$$

$$\begin{aligned} &= \mu'(R \mid \mathbf{1}) \cdot \mu(\mathbf{1}) + \tfrac{\alpha}{1-\alpha/2} \cdot \mu'(\overline{R} \mid \mathbf{0}) \cdot \mu(\mathbf{0}) \\ &\geq \alpha \cdot \mu(\mathbf{1}) + \tfrac{\alpha}{1-\alpha/2} \cdot (1-\alpha/2) \cdot \mu(\mathbf{0}) \\ &= \alpha. \end{aligned}$$

Now by the minimax theorem, there exists a distribution D over rectangles such that for every input (x, y), $\mathbb{E}(M_{(x,y),D}) \geq \alpha$. If $f(x, y) = 1$ this means the probability a random rectangle from D contains (x, y) is at least α. If $f(x, y) = 0$ this means $\frac{\alpha}{1-\alpha/2}$ times the probability a random rectangle from D does not contain (x, y) is at least α, in other words the probability a random rectangle from D contains (x, y) is at most $\alpha/2$. Thus the protocol that picks a random rectangle from D and accepts iff the input is in the rectangle shows that $\mathsf{R}^{\mathsf{pub}}_{\alpha,\alpha/2}(f) \leq 2$ and hence $\mathsf{SBP}(f) \leq 2 + \log(1/\alpha) = 2 + \mathsf{Corr}(f)$.

3 USBP Lower Bound

In this section we prove Theorem 3, which states that $\mathsf{USBP}(\textsc{Disj}) = \Theta(n)$. We first give an informal overview.

Our proof uses the by-now standard information complexity approach [4, 12]. In this approach, one considers some suitably distributed random input (X, Y) and measures the amount of information that the protocol transcript $\Pi(X, Y)$ (i.e., the concatenation of all messages sent) "leaks" about the input as quantified by the *mutual information* $\mathbb{I}(\Pi(X, Y); X, Y)$. Lower bounding the mutual information has the side effect of lower bounding the *entropy* $\mathbb{H}(\Pi(X, Y))$ of the transcript, which in turn lower bounds the length of the transcript and thereby the communication complexity. It is often useful to involve the *conditional* versions of these information measures, defined by $\mathbb{H}(\Pi \mid Z) = \mathbb{E}_{z\sim Z}\mathbb{H}(\Pi \mid Z = z)$ and $\mathbb{I}(\Pi; X, Y \mid Z) = \mathbb{E}_{z\sim Z}\mathbb{I}(\Pi; X, Y \mid Z = z)$ where Z is some random variable (jointly distributed with X and Y). We refer the reader to [14] for discussions of these basic information theory concepts.

A key benefit of studying mutual information is that one automatically obtains for it a *direct sum* property (as in [12, 4]), as long as the coordinates (X_i, Y_i), $i \in [n]$, are mutually independent. This way, the task of proving an $\Omega(n)$ lower bound for the original problem reduces to the task of proving an $\Omega(1)$ information lower bound for some constant-size "gadget". For set-disjointness $\textsc{Disj} = \textsc{And}_n \circ \textsc{Nand}^n$ this gadget is typically $\textsc{Nand}$.

Our proof follows this outline. The reduction to the single-gadget case will be packaged into Theorem 8 and is standard. By contrast, in proving the $\Omega(1)$ information lower bound for the single gadget, we need to overcome the following two new technical issues.

(1) Small Acceptance Probabilities. Since the protocol is only required to succeed with a tiny probability $\alpha(n)$ on 1-inputs, the transcript of Π can be useless most of the time: Imagine a protocol that rejects with probability $1 - \alpha$ at the start (and otherwise does something useful). The entropy of the transcript of such protocols can be as low as $O(\alpha)$.

To address this issue, we do not work with the transcript distribution of Π directly, but rather with the *conditional distribution* given that the protocol accepts. That is, for 1-inputs (x, y), we consider the random variable

$$T(x, y) \coloneqq \Pi(x, y) \mid \Pi(x, y) \text{ is an accepting transcript}$$

and proceed to lower bound $\mathbb{I}(T(X, Y); X, Y)$ instead. One subtlety is that conditioning on acceptance does not "commute" with the reduction to the single-gadget case. We must

Protocol Π^*. On input $(x, y) \in \{0,1\}^2$:

1. If $x = 0$ Alice sends a "1". If $x = 1$ Alice sends a "1" with probability α and rejects otherwise (by sending a "0").
2. Suppose Alice sent a "1". Then if $y = 0$ Bob accepts (by sending a "1"). If $y = 1$ then Bob accepts with probability α and rejects otherwise.

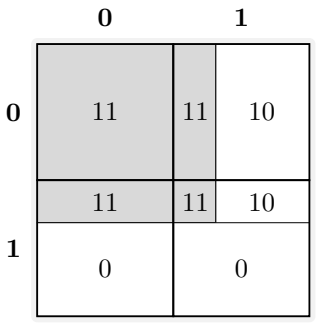


Figure 1 Protocol Π^* for NAND. In the illustration on the right, each of the input blocks is further subdivided into rectangles according to the outcomes of the private coins. The rectangles are labeled with the associated transcripts.

consider the distribution of T that arises from first conditioning on acceptance and then doing the reduction, which is generally not the same distribution as if we did the reduction and then conditioned on acceptance. However, this is not a significant technical obstacle.

(2) Large Acceptance Probabilities. The acceptance probability of a protocol Π can vary between α and 1 when run on different 1-inputs. This, together with our conditioning idea above, introduces a new problem: there are USBP protocols for NAND such that the associated T *leaks no information about the input!*

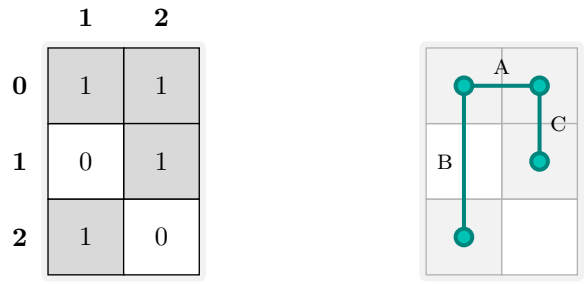
Indeed, consider the protocol Π^* for NAND given in Figure 1. This protocol accepts the 1-input $(0,0)$ with probability 1, the 1-inputs $(0,1)$ and $(1,0)$ with probability α, and the 0-input $(1,1)$ with probability α^2. Choosing α such that $\alpha^2 \leq \alpha/2$ we obtain a USBP protocol for NAND where the associated conditioned-on-acceptance variable T^* is constant (the protocol Π^* has only one accepting transcript, namely "11").

To avoid this problem, we use a more complicated gadget than NAND; see Figure 2a. The new gadget G contains two instances of NAND: in Figure 2b one instance of NAND corresponds to the pair of edges AB and another one to AC. We show that the bad situation described above cannot happen simultaneously for both of them. One subtlety is that the bad situation—i.e., when a transcript has much higher probability of appearing on the 1-input $(0,0)$ of NAND than on the other two 1-inputs $(0,1)$ and $(1,0)$ of NAND—depends on the transcript, with some transcripts being bad for AB and some being bad for AC, but none being bad for both. We prove an information lower bound (conditioned on acceptance) for whichever instance of NAND behaves better for "most" transcripts.

We note that a similar technical issue arose in the proof of Braverman and Moitra [6] when analyzing the case $\alpha = 1/2 + \epsilon$, $\beta = 1/2 - \epsilon$. Their solution involved applying a certain type of *random-self-reduction* (they called it *smoothing*) to the inputs before invoking the protocol. This approach is highly tailored to their setting and does not seem to be directly helpful to us. Nevertheless, our gadget G was inspired by their analysis.

3.1 Proof of Theorem 3

Define the gadget $G\colon \{0,1,2\} \times \{1,2\} \to \{0,1\}$ as the indicator for non-equality; see Figure 2a. Define the function $F\colon \{0,1,2\}^n \times \{1,2\}^n \to \{0,1\}$ by $F = \text{AND}_n \circ G^n$, i.e., $F(x,y) = 1$ iff $G(x_i, y_i) = 1$ for all $i \in [n]$. Since G reduces to DISJ on 2 bits by the map



(a) (b)

Figure 2 (a) Truth table of the gadget G. (b) Distributions Q_{A}, Q_{B}, and Q_{C} are uniform over the endpoints of the edges A, B, and C, respectively.

$(0 \mapsto 00,\ 1 \mapsto 01,\ 2 \mapsto 10)$, we find that F reduces to Disj on $2n$ bits. Hence it suffices to prove that $\mathsf{USBP}(F) \geq \Omega(n)$.

Input Distribution. Define $Q_{\text{A}}, Q_{\text{B}}, Q_{\text{C}}$ to be the following three distributions over $\{0,1,2\} \times \{1,2\}$: Q_{A} is uniform over $\{(0,1),(0,2)\}$, Q_{B} is uniform over $\{(0,1),(2,1)\}$, and Q_{C} is uniform over $\{(0,2),(1,2)\}$; see Figure 2b. For $i \in [n]$, $u \in \{0,1,2\}$, $v \in \{1,2\}$, and z a length-$(n-1)$ string over the alphabet $\{\text{A},\text{B},\text{C}\}$ indexed by $[n]\backslash\{i\}$, define $D_{i,u,v,z}$ to be the distribution over pairs $(x,y) \in \{0,1,2\}^n \times \{1,2\}^n$ obtained by setting $x_i = u$, $y_i = v$, and for each $j \neq i$ (independently) sampling (x_j, y_j) from Q_{z_j}. Note that $\text{support}(D_{i,u,v,z}) \subseteq F^{-1}(G(u,v))$ and that x and y are independent when sampled from $D_{i,u,v,z}$.

Reduction to the Single-gadget Case. Let Π be an $\mathsf{R}^{\mathsf{priv}}_{\alpha,\alpha/4}$ protocol for F (recall that by and-amplification we may assume $\beta = \alpha/4$ in the definition of USBP). Let $\Pi(x,y)$ denote the transcript of Π on input (x,y). Thus $\Pi(x,y)$ is a random variable whose outcome depends on the private coins of the protocol. For $(x,y) \in F^{-1}(1)$ define $T(x,y)$ as the random variable whose distribution is that of $\Pi(x,y)$ conditioned on $\Pi(x,y)$ being an accepting transcript.

Suppose for contradiction that the transcripts have length less than $n/2400$. Using the direct sum methodology we will next find a coordinate i (and a string z) such that the protocol leaks very little information about the i-th input (conditioned on the data z). This is formalized in the following lemma whose proof we defer to subsection 3.2 as it is essentially identical to the corresponding argument in [4]. Below, $\|X - Y\|$ denotes the statistical distance between the distributions of the random variables X and Y.

▶ **Lemma 8.** *If $\gamma > 0$ is such that for all i and z either $\|T(D_{i,0,1,z}) - T(D_{i,0,2,z})\| \geq \gamma$ or $\|T(D_{i,0,1,z}) - T(D_{i,2,1,z})\| \geq \gamma$ or $\|T(D_{i,0,2,z}) - T(D_{i,1,2,z})\| \geq \gamma$, then some transcript has length at least $n\gamma^2/6$.*

Contrapositively, letting $\gamma = 1/20$, Theorem 8 implies that there exists an i and z (which we fix henceforth) such that $\|T(D_{i,0,1,z}) - T(D_{i,0,2,z})\|$, $\|T(D_{i,0,1,z}) - T(D_{i,2,1,z})\|$, and $\|T(D_{i,0,2,z}) - T(D_{i,1,2,z})\|$ are all less than γ.

The Single-gadget Case. Let (u,v) be an input to G and let τ be a transcript. We define

$$\pi_{uv}(\tau) \coloneqq \Pr\big[\Pi(D_{i,u,v,z}) = \tau\big] \quad \text{for any } (u,v),$$
$$t_{uv}(\tau) \coloneqq \Pr\big[T(D_{i,u,v,z}) = \tau\big] \quad \text{for } (u,v) \in G^{-1}(1).$$

We henceforth adopt the convention that $0/0 = 0$. Let

$$S \coloneqq \left\{\text{accepting } \tau \;:\; \frac{\pi_{01}(\tau)}{t_{01}(\tau)} \leq \frac{\pi_{02}(\tau)}{t_{02}(\tau)}\right\}$$

and let $\overline{S} \coloneqq \{\text{accepting } \tau\} \backslash S$. Since $\|T(D_{i,0,1,z}) - T(D_{i,0,2,z})\| < \gamma$, we have either $\Pr\big[T(D_{i,0,1,z}) \in S\big] \geq \frac{1-\gamma}{2}$ or $\Pr\big[T(D_{i,0,2,z}) \in \overline{S}\big] \geq \frac{1-\gamma}{2}$. Henceforth assume the former case; a completely analogous argument handles the latter case.

Note that $\pi_{22}(\tau) \cdot \pi_{01}(\tau) = \pi_{21}(\tau) \cdot \pi_{02}(\tau)$ by the basic rectangular structure of τ. Also note that if $G(u,v) = 1$ and τ is accepting, then the following both hold.

- We have $\pi_{uv}(\tau) = 0$ iff $t_{uv}(\tau) = 0$, and hence $\pi_{uv}(\tau) = \frac{\pi_{uv}(\tau)}{t_{uv}(\tau)} \cdot t_{uv}(\tau)$.
- Assuming $\pi_{uv}(\tau)$ and $t_{uv}(\tau)$ are nonzero, we have $\frac{\pi_{uv}(\tau)}{t_{uv}(\tau)} \geq \alpha$ by the correctness of Π.

For $(u,v) \in \{(2,1),(0,2)\}$ define $\gamma_{uv}(\tau) = |t_{uv}(\tau) - t_{01}(\tau)|$ and note that

$$\textstyle\sum_{\text{accepting } \tau} \gamma_{uv}(\tau) = 2\|T(D_{i,u,v,z}) - T(D_{i,0,1,z})\| < 2\gamma. \tag{3}$$

By a case analysis, we have $t_{21}(\tau) \cdot t_{02}(\tau) \geq t_{01}(\tau)^2 - t_{01}(\tau)\big(\gamma_{21}(\tau) + \gamma_{02}(\tau)\big)$. Recalling our convention that $0/0 = 0$, and considering the case $t_{01}(\tau) = 0$ separately, we find that in all cases

$$\frac{t_{21}(\tau) \cdot t_{02}(\tau)}{t_{01}(\tau)} \geq t_{01}(\tau) - \gamma_{21}(\tau) - \gamma_{02}(\tau). \tag{4}$$

Thus we have

$$\begin{aligned}
\Pr\big[\Pi(D_{i,2,2,z}) \text{ accepts}\big] &= \sum_{\text{accepting } \tau} \pi_{22}(\tau) \\
&\geq \sum_{\tau \in S} \pi_{22}(\tau) \\
&\geq \sum_{\tau \in S} \frac{\pi_{21}(\tau) \cdot \pi_{02}(\tau)}{\pi_{01}(\tau)} \\
&= \sum_{\tau \in S} \frac{\frac{\pi_{21}(\tau)}{t_{21}(\tau)} \cdot \frac{\pi_{02}(\tau)}{t_{02}(\tau)}}{\frac{\pi_{01}(\tau)}{t_{01}(\tau)}} \cdot \frac{t_{21}(\tau) \cdot t_{02}(\tau)}{t_{01}(\tau)} \\
&\geq \sum_{\tau \in S} \alpha \cdot \big(t_{01}(\tau) - \gamma_{21}(\tau) - \gamma_{02}(\tau)\big) \\
&> \alpha \cdot \big(\tfrac{1-\gamma}{2} - 2\gamma - 2\gamma\big) \\
&> \alpha/4.
\end{aligned}$$

To see that the fifth line follows from the fourth, consider each $\tau \in S$: If $t_{21}(\tau) \neq 0$ and $t_{02}(\tau) \neq 0$ then it follows by $\frac{\pi_{21}(\tau)}{t_{21}(\tau)} \geq \alpha$ and $\frac{\pi_{02}(\tau)}{t_{02}(\tau)} \Big/ \frac{\pi_{01}(\tau)}{t_{01}(\tau)} \geq 1$ (since $\tau \in S$) and (4). On the other hand, if $t_{21}(\tau) = 0$ or $t_{02}(\tau) = 0$, say, $t_{21}(\tau) = 0$, then it follows since the summand on the fourth line is 0, and $t_{01}(\tau) - \gamma_{21}(\tau) = 0$ so the summand on the fifth line is nonpositive. The sixth line follows from the fifth by $\sum_{\tau \in S} t_{01}(\tau) = \Pr\big[T(D_{i,0,1,z}) \in S\big] \geq \frac{1-\gamma}{2}$ and $\sum_{\tau \in S} \gamma_{21}(\tau) \leq \sum_{\text{accepting } \tau} \gamma_{21}(\tau)$ and (3), and similarly for γ_{02}.

We conclude that $\Pr\big[\Pi(x,y) \text{ accepts}\big] > \alpha/4$ for some $(x,y) \in \text{support}(D_{i,2,2,z}) \subseteq F^{-1}(0)$, contradicting the correctness of Π. This finishes the proof of Theorem 3.

3.2 Proof of Theorem 8

Define jointly distributed random variables $X = X_1 \cdots X_n \in \{0,1,2\}^n$, $Y = Y_1 \cdots Y_n \in \{1,2\}^n$, and $Z = Z_1 \cdots Z_n \in \{\text{A}, \text{B}, \text{C}\}^n$ as follows: Z is uniform, and given a particular

choice of Z, for each $i \in [n]$ (independently) (X_i, Y_i) is sampled from Q_{Z_i}. Thus the marginal distribution of (X, Y) is that for each i (independently), (X_i, Y_i) has probability $1/3$ for each of $(0,1),(0,2)$, and probability $1/6$ for each of $(2,1),(1,2)$. Since the support of (X, Y) is in $F^{-1}(1)$, we may also view T as a random variable distributed jointly with (X, Y, Z). Let Z_{-i} denote $Z_1 \cdots Z_{i-1} Z_{i+1} \cdots Z_n$.

For any $i \in [n]$, $z_i \in \{\text{A}, \text{B}, \text{C}\}$, and $z_{-i} \in \{\text{A}, \text{B}, \text{C}\}^{n-1}$ indexed by $[n]\backslash\{i\}$, we can view $\big(T(D_{i,Q_{z_i},z_{-i}}), Q_{z_i}\big)$ as a pair of jointly distributed random variables that is distributed identically to $\big(T, (X_i, Y_i) \;\big|\; Z_i = z_i, Z_{-i} = z_{-i}\big)$. For all i and z_{-i}, by a standard lemma (see [4, Lemma 6.2 and Proposition 6.10]) we have $\mathbb{I}\big(T(D_{i,Q_\text{A},z_{-i}}); Q_\text{A}\big) \geq \|T(D_{i,0,1,z_{-i}}) - T(D_{i,0,2,z_{-i}})\|^2/2$ and similarly for B and C. Therefore

$$\begin{aligned}
\text{Maximum length of transcript} &\geq \mathbb{H}(T \mid Z) \\
&\geq \mathbb{I}(T; X, Y \mid Z) \\
&\geq \textstyle\sum_i \mathbb{I}(T; X_i, Y_i \mid Z) \\
&= \textstyle\sum_i \mathbb{E}_{z_{-i}} \frac{1}{3} \sum_{z_i} \mathbb{I}\big(T; X_i, Y_i \;\big|\; Z_i = z_i, Z_{-i} = z_{-i}\big) \\
&= \textstyle\sum_i \mathbb{E}_{z_{-i}} \frac{1}{3} \sum_{z_i} \mathbb{I}\big(T(D_{i,Q_{z_i},z_{-i}}); Q_{z_i}\big) \\
&\geq \textstyle\sum_i \mathbb{E}_{z_{-i}} \frac{1}{3} \cdot \frac{1}{2} \Big(\|T(D_{i,0,1,z_{-i}}) - T(D_{i,0,2,z_{-i}})\|^2 \\
&\qquad\qquad + \|T(D_{i,0,1,z_{-i}}) - T(D_{i,2,1,z_{-i}})\|^2 \\
&\qquad\qquad + \|T(D_{i,0,2,z_{-i}}) - T(D_{i,1,2,z_{-i}})\|^2 \Big) \\
&\geq \textstyle\sum_i \mathbb{E}_{z_{-i}} (\gamma^2/6) \\
&= n\gamma^2/6.
\end{aligned}$$

where the third line follows by a standard direct sum property for conditional information cost [4].

4 The Complexity of Set-Disjointness

We now prove Theorem 4 and Theorem 5 using Theorem 3 and Theorem 2.

4.1 Lower Bounds

Private-coin Lower Bounds. Let Π be an $\mathsf{R}^{\mathsf{priv}}_{\alpha,\beta}$ protocol for Disj. We prove that the cost of Π is both $\Omega(n \cdot (1 - \beta/\alpha))$ and $\Omega(\log n)$, as required for Theorem 4.

First, if we do and-amplification by iterating the protocol $\lceil 1/(1 - \beta/\alpha)\rceil$ times and accepting iff all runs accept, we get an $\mathsf{R}^{\mathsf{priv}}_{\alpha',\alpha'/2}$ protocol for Disj with $\alpha' = \alpha^{\lceil 1/(1-\beta/\alpha)\rceil}$ (since $(\beta/\alpha)^{\lceil 1/(1-\beta/\alpha)\rceil} < 1/2$). By Theorem 3 the amplified protocol must use $\Omega(n)$ communication and hence Π must have used $\Omega(n \cdot (1 - \beta/\alpha))$ communication.

Second, Forster's result [16] that the UPP complexity of inner product is $\Omega(n)$ gives us the $\Omega(\log n)$ lower bound for Π. Indeed, the inner product function reduces to Disj with exponential blow-up (see [35, Proposition 6.5]) and we may convert Π into an UPP protocol by shifting the acceptance threshold near $1/2$.

Public-coin Lower Bounds. Let Π be an $\mathsf{R}^{\mathsf{pub}}_{\alpha,\beta}$ protocol for Disj. We consider the two parts of Theorem 5 separately.

For the first part, suppose $\log(1/\alpha) \leq C \cdot n \cdot (1 - \beta/\alpha)$ for a to-be-specified constant C. We proceed exactly as above: We first and-amplify Π into an $\mathsf{R}^{\mathsf{priv}}_{\alpha',\alpha'/2}$ protocol. The parameters satisfy $\log(1/\alpha') = \log(1/\alpha) \cdot \lceil 1/(1-\beta/\alpha)\rceil \leq C \cdot n \cdot (1-\beta/\alpha) \cdot \lceil 1/(1-\beta/\alpha)\rceil \leq 2C \cdot n$. Hence

if C is a sufficiently small universal constant then the $\Omega(n)$ lower bound for the amplified protocol (provided now by Theorem 2) must be coming from the communication cost and not from the $\log(1/\alpha')$ term. We conclude that the original protocol Π must have used $\Omega(n \cdot (1-\alpha/\beta))$ communication.

For the second part, we do not need any restriction on the parameters. We claim that since DISJ has a 2×2 identity submatrix, we cannot have $\mathsf{R}^{\mathsf{pub}}_{\alpha,\beta}(\text{DISJ}) \leq 1$.[1] Suppose for contradiction there is a 1-bit protocol and yet $\text{DISJ}(x,y) = \text{DISJ}(x',y') = 1$ and $\text{DISJ}(x,y') = \text{DISJ}(x',y) = 0$. Say r is the probability Alice declares the output and $1-r$ is the probability Bob declares the output. Conditioned on Alice declaring the output let $p_x, p_{x'}$ be the acceptance probability for the x and x' rows, and conditioned on Bob declaring the output let $q_y, q_{y'}$ be the acceptance probability for the y and y' columns. Letting $\pi_{xy} = rp_x + (1-r)q_y$ be the overall acceptance probability on input (x,y), we have $\alpha - \beta \leq \pi_{xy} - \pi_{x'y} = r(p_x - p_{x'})$ and $\alpha - \beta \leq \pi_{x'y'} - \pi_{xy'} = r(p_{x'} - p_x)$, a contradiction.

4.2 Upper Bounds

Public-coin Protocols. We start with a simple $\mathsf{R}^{\mathsf{pub}}_{1,\beta/\alpha}$ protocol for DISJ of cost $\Theta(n \cdot (1-\beta/\alpha))$.

Basic public-coin protocol Π.
1. Use public randomness to pick a uniformly random $S \subseteq [n]$ of size $\lceil n \cdot (1-\beta/\alpha) \rceil$.
2. Alice sends the substring $x|_S$ to Bob.
3. Bob outputs $\text{DISJ}(x|_S, y|_S)$.

It is straightforward to check that Π is indeed an $\mathsf{R}^{\mathsf{pub}}_{1,\beta/\alpha}$ protocol. To obtain an $\mathsf{R}^{\mathsf{pub}}_{\alpha,\beta}$ protocol for the first part of Theorem 5 (without needing any restriction on the parameters), we can reject with probability $1-\alpha$ at the beginning and otherwise run Π. To obtain a protocol of cost 2 for the second part of Theorem 5, we need to better exploit the power of public coins. If we modify Π so that additional public coins are used to guess $x|_S$, then Alice can just send one bit indicating whether the guess is correct, and Bob can send the output bit (rejecting if the guess was wrong). This yields an $\mathsf{R}^{\mathsf{pub}}_{1/2^{|S|},\,\beta/\alpha 2^{|S|}}$ protocol which, by the restriction that $\alpha \leq 1/2^{|S|}$, can be adapted into an $\mathsf{R}^{\mathsf{pub}}_{\alpha,\beta}$ protocol by automatically rejecting with probability $1-\alpha 2^{|S|}$.

In fact, the above protocols can be seen as special cases of the following general protocol, which interpolates between them. For simplicity of presentation, let us assume that $\log(1/\alpha)$ is an integer and $\log(1/\alpha) \leq |S|$. In step 2 of the basic protocol Π, Alice can expedite the sending of her message to Bob as follows: Alice and Bob interpret additional public coins as guessing the first $\log(1/\alpha)$ bits of Alice's message. Alice can use one bit of communication to indicate whether this guess is correct, and if so she can send the other $|S| - \log(1/\alpha)$ bits

[1] For the public-coin version of UPP, an equivalence actually holds. For all f, we have $\mathsf{UPP}^{\mathsf{pub}}(f) \leq 2$, and it is not difficult to show that the following are equivalent: (i) $\mathsf{UPP}^{\mathsf{pub}}(f) \leq 1$, (ii) there exist row and column values $p_x, q_y \in [0,1]$ and $r \in [0,1]$ such that $\left|rp_x + (1-r)q_y - f(x,y)\right| < 1/2$, (iii) the rows and columns can be permuted so each row and each column is monotonically nondecreasing (0's then 1's), (iv) f does not contain as a submatrix the 2×2 identity (or its complement). To see that (iii)⇒(ii), take $r = 1/2$, and p_x = fraction of 1's in the x row, and $q_y = (y - 1/2)/(\text{number of columns})$ where y is viewed as a positive integer.

of her message normally. The probability that the public guess is correct is $2^{-\log(1/\alpha)} = \alpha$. Thus, this new protocol ends up working in a familiar way: with probability $1-\alpha$ the public guess fails (in which case we reject), but otherwise we are able to run Π successfully. This results in an $\mathsf{R}^{\mathsf{pub}}_{\alpha,\beta}$ protocol of cost $|S| - \log(1/\alpha) + 2$. Here the $+2$ comes from Alice indicating whether the public guess is correct and Bob sending the final answer.

Private-coin Protocols. By sparsification, we may assume the basic protocol Π uses only $O(\log n)$ bits of public randomness. Thus we have $\mathsf{R}^{\mathsf{priv}}_{1,\beta/\alpha}(\textsc{Disj}) \leq O(n \cdot (1-\beta/\alpha) + \log n)$ since Alice can pick S privately and send it to Bob along with $x|_S$. An $\mathsf{R}^{\mathsf{priv}}_{\alpha,\beta}$ protocol for Theorem 4 can be obtained as previously: automatically reject with probability $1-\alpha$ and otherwise run the $\mathsf{R}^{\mathsf{priv}}_{1,\beta/\alpha}$ protocol.

5 Open Problems

It would be interesting to separate SBP and USBP, or to separate MA and SBP, even by a promise problem. In the classical world, it is known that MA and SBP can be separated by an oracle [5, 32]. The relationship between USBP and AM is also open (in both directions).

Among the complexity measures PP, UPP, SBP, and USBP, the first three are characterized by discrepancy [26], sign-rank [30], and corruption (Theorem 1), respectively. It is straightforward to show that the fourth is characterized by the log of the smallest nonnegative rank of a matrix such that the minimum value of a 1-input's entry is at least twice the maximum value of a 0-input's entry. It is open to provide a more natural characterization of USBP.

Acknowledgements. We thank Mark Braverman, Tom Gur, Toniann Pitassi, and anonymous reviewers for comments and discussions.

References

1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Transactions on Computation Theory*, 1(1), 2009.

2 László Babai, Peter Frankl, and Janos Simon. Complexity classes in communication complexity theory. In *Proceedings of the 27th Symposium on Foundations of Computer Science (FOCS)*, pages 337–347. IEEE, 1986.

3 László Babai and Shlomo Moran. Arthur–Merlin games: A randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36(2):254–276, 1988.

4 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.

5 Elmar Böhler, Christian Glaßer, and Daniel Meister. Error-bounded probabilistic computations between MA and AM. *Journal of Computer and System Sciences*, 72(6):1043–1076, 2006.

6 Mark Braverman and Ankur Moitra. An information complexity approach to extended formulations. In *Proceedings of the 45th Symposium on Theory of Computing (STOC)*, pages 161–170. ACM, 2013.

7 Mark Braverman and Omri Weinstein. A discrepancy lower bound for information complexity. In *Proceedings of the 16th International Workshop on Randomization and Computation (RANDOM)*, pages 459–470. Springer, 2012.

8 Harry Buhrman, Nikolai Vereshchagin, and Ronald de Wolf. On computation and communication with small bias. In *Proceedings of the 22nd Conference on Computational Complexity (CCC)*, pages 24–32. IEEE, 2007.

9 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. Annotations in data streams. In *Proceedings of the 36th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 222–234. Springer, 2009.

10 Amit Chakrabarti, Graham Cormode, Andrew McGregor, Justin Thaler, and Suresh Venkatasubramanian. On interactivity in Arthur–Merlin communication and stream computation. Technical Report TR13-180, Electronic Colloquium on Computational Complexity (ECCC), 2013.

11 Amit Chakrabarti and Oded Regev. An optimal lower bound on the communication complexity of Gap-Hamming-Distance. *SIAM Journal on Computing*, 41(5):1299–1317, 2012.

12 Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *Proceedings of the 42nd Symposium on Foundations of Computer Science (FOCS)*, pages 270–278. IEEE, 2001.

13 Arkadev Chattopadhyay and Toniann Pitassi. The story of set disjointness. *SIGACT News*, 41(3):59–85, 2010.

14 Thomas Cover and Joy Thomas. *Elements of Information Theory*. Wiley, 2006.

15 Scott Diehl. Lower bounds for swapping Arthur and Merlin. In *Proceedings of the 11th International Workshop on Randomization and Computation (RANDOM)*, pages 449–463. Springer, 2007.

16 Jürgen Forster. A linear lower bound on the unbounded error probabilistic communication complexity. *Journal of Computer and System Sciences*, 65(4):612–625, 2002.

17 Anat Ganor, Gillat Kol, and Ran Raz. Exponential separation of information and communication. Technical Report TR14-049, Electronic Colloquium on Computational Complexity (ECCC), 2014.

18 Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the 18th Symposium on Theory of Computing (STOC)*, pages 59–68. ACM, 1986.

19 Tom Gur and Ran Raz. Arthur–Merlin streaming complexity. In *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 528–539. Springer, 2013.

20 Rahul Jain and Hartmut Klauck. The partition bound for classical communication complexity and query complexity. In *Proceedings of the 25th Conference on Computational Complexity (CCC)*, pages 247–258. IEEE, 2010.

21 Rahul Jain, Troy Lee, and Nisheeth Vishnoi. A quadratically tight partition bound for classical communication complexity and query complexity. Technical report, arXiv, 2014.

22 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*, volume 27 of *Algorithms and Combinatorics*. Springer, 2012.

23 Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM Journal on Discrete Mathematics*, 5(4):545–557, 1992.

24 Iordanis Kerenidis, Sophie Laplante, Virginie Lerays, Jérémie Roland, and David Xiao. Lower bounds on information complexity via zero-communication protocols and applications. In *Proceedings of the 53rd Symposium on Foundations of Computer Science (FOCS)*, pages 500–509, 2012.

25 Hartmut Klauck. Rectangle size bounds and threshold covers in communication complexity. In *Proceedings of the 18th Conference on Computational Complexity (CCC)*, pages 118–134. IEEE, 2003.

26 Hartmut Klauck. Lower bounds for quantum communication complexity. *SIAM Journal on Computing*, 37(1):20–46, 2007.

27 Hartmut Klauck. On Arthur Merlin games in communication complexity. In *Proceedings of the 26th Conference on Computational Complexity (CCC)*, pages 189–199. IEEE, 2011.

28 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.

29 Ilan Newman. Private vs. common random bits in communication complexity. *Information Processing Letters*, 39(2):67–71, 1991.

30 Ramamohan Paturi and Janos Simon. Probabilistic communication complexity. *Journal of Computer and System Sciences*, 33(1):106–123, 1986.

31 Alexander Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992.

32 Miklos Santha. Relativized Arthur–Merlin versus Merlin–Arthur games. *Information and Computation*, 80(1):44–49, 1989.

33 Alexander Sherstov. Halfspace matrices. *Computational Complexity*, 17(2):149–178, 2008.

34 Alexander Sherstov. The pattern matrix method. *SIAM Journal on Computing*, 40(6):1969–2000, 2011.

35 Alexander Sherstov. The unbounded-error communication complexity of symmetric functions. *Combinatorica*, 31(5):583–614, 2011.

36 Alexander Sherstov. The communication complexity of Gap Hamming Distance. *Theory of Computing*, 8(1):197–208, 2012.

37 Thomas Vidick. A concentration inequality for the overlap of a vector on a large set, with application to the communication complexity of the Gap-Hamming-Distance problem. *Chicago Journal of Theoretical Computer Science*, 2012(1):1–12, 2012.

38 Thomas Watson. The complexity of estimating min-entropy. *Computational Complexity*, 2015. To appear. Preprint: `http://eccc.hpi-web.de/report/2012/070/`.

List Decoding Group Homomorphisms Between Supersolvable Groups*

Alan Guo[1] **and Madhu Sudan**[2]

1 **CSAIL, Massachusetts Institute of Technology**
32 Vassar Street, Cambridge, MA, USA
aguo@mit.edu

2 **Microsoft Research**
One Memorial Drive, Cambridge, MA, USA
madhu@mit.edu

Abstract

We show that the set of homomorphisms between two supersolvable groups can be locally list decoded up to the minimum distance of the code, extending the results of Dinur et al who studied the case where the groups are abelian. Moreover, when specialized to the abelian case, our proof is more streamlined and gives a better constant in the exponent of the list size. The constant is improved from about 3.5 million to 105.

1998 ACM Subject Classification F.2.2 Computation on Discrete Structures

Keywords and phrases Group theory, error-correcting codes, locally decodable codes

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.737

1 Introduction

It is well-known that for any pair of groups G and H with G being finite, the set of homomorphisms from G to H form an error-correcting code with $\omega(1)$ distance (since any two distinct homomorphisms agree on a subgroup of G which has size a constant factor smaller than that of G). The most classical example of such a setting is when G is the additive group over $\mathbb{F}_2^n$ and $H = \mathbb{F}_2$ (where $\mathbb{F}_q$ denotes the finite field of size q). The seminal work of Goldreich and Levin [3] gave an "efficient local list-decoding" algorithm for this particular setting. Such an algorithm has oracle access to a function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, and given $\epsilon > 0$, reports all homomorphisms ϕ that agree with f on $1/2 + \epsilon$ fraction of the points in time $\mathrm{poly}(\log |G|, \log |H|, 1/\epsilon)$.

A natural question, given the centrality of the Goldreich-Levin algorithm in coding theory and learning theory, is to ask what is the most general setting in which it works. In particular, one abstraction of the (original) Goldreich-Levin algorithm is that it uses coding theory (in particular, the Johnson bound of coding theory) to get a combinatorial bound on the list size, namely the number of functions that may have agreement $1/2 + \epsilon$ with the function f. It then uses some decomposability properties of the domain $\mathbb{F}_2^n$ to get an algorithm for the list-decoding. Grigorescu et al. [7] and Dinur et al. [1], extended this abstraction to the more general setting of abelian groups. They first analyze $\delta_{G,H}$, the minimum possible distance between two homomorphisms from G to H. They then consider the task of recovering all homomorphisms at distance $\delta_{G,H} - \epsilon$ from a given function f. Roughly they show that the

* The first author was partially supported by NSF grants CCF-0829672, CCF-1065125, and CCF-6922462, and an NSF Graduate Research Fellowship.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 737–747

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

"decomposability" used in the algorithmic step of Goldreich and Levin can be generalized to the case of direct sum of abelian groups, so if $G = G_1 \oplus G_2 \oplus \cdots \oplus G_k$ and each G_i is small and also if H is small, then the algorithmic step can be extended. This reduces the list-decoding question to the combinatorial one. Here the standard bounds from coding theory are insufficient, however one can use decompositions of the group H into prime cyclic groups to show that the list size is at most $\text{poly}(1/\epsilon)$.

In this work, we take this line of work a step further and explore this algorithm in the setting where G and H are not abelian. In this setting decompositions of G and H turn out to be more complex, and indeed even the question of determining $\delta_{G,H}$ turns out to be non-trivial. This question is explored in a companion work by the first author [8], where $\delta_{G,H}$ is determined explicitly for a broad class of groups, including the case of "supersolvable" groups which we study here. To describe the groups we consider we recall some basic group-theoretic terminology.

A subset $N \subseteq G$ is a subgroup of G, denote $N \leq G$, if N is closed under the group operation. A subgroup $N \leq G$ is said to be normal in G, denoted $N \triangleleft G$, if $aN = Na$ for all $a \in G$, where $aN = \{an | n \in N\}$ and $Na = \{na | n \in N\}$. If $N \triangleleft G$, then the set of cosets of N in G form a group under the operation $(aN)(bN) = (abN)$. This group is denoted G/N. G is *solvable* if there exists a series of groups $\{1_G\} = G_0 \triangleleft G_1 \triangleleft \cdots \triangleleft G_k = G$ such that G_i/G_{i-1} is abelian for every i. We refer to the sequence $\langle 1_G = G_0, G_1, \ldots, G_k = G\rangle$ as the solvability chain of G. G is *supersolvable* if it has a solvability chain $\langle 1_G = G_0, G_1, \ldots, G_k = G\rangle$ where $G_i \triangleleft G$ and G_i/G_{i-1} is cyclic for every i.

1.1 Our Results

Our main results, stated somewhat informally, are the following:

- **(Combinatorial list decodability)** There exists a constant $C \approx 105$ such that if G and H are *supersolvable* groups, then for any $f : G \to H$, the number of (affine) homomorphisms from G to H disagreeing with f on less than $\delta_{G,H} - \epsilon$ fraction of G is at most $(1/\epsilon)^C$. (See Theorem 3.4.)
- **(Algorithmic list decodability)** Let G be a solvable group and H be any group such that the set of homomorphisms from G to H have nice combinatorial list-decodability, i.e., the number of homomorphisms from G to H that have distance $\delta_{G,H} - \epsilon$ from a fixed function f is at most $(1/\epsilon)^C$. Then, the set of homomorphisms from G to H can be locally list decoded up to $\delta_{G,H} - \epsilon$ errors in $\text{poly}(\log|G|, \log|H|, \frac{1}{\epsilon})$ time assuming oracle access to the multiplication table of H.[1] (See Theorem 4.2.)

Putting the two ingredients together we get efficient list-decoding algorithms up to radius $\delta_{G,H} - \epsilon$ whenever G and H are supersolvable.

Potential Extensions and Limits

The case of solvable groups appears to be a natural limit to the nature of results given above, but we are not able achieve even this limit due to technical limitations which only allows us to deal with the case where the quotient group of successive members in the solvability chain are cyclic. It seems possible to go slightly beyond the results mentioned above though.

[1] For the group G we only need to be able sample its elements in a specific way, and compute f on elements sampled in such a way. Using the (super)solvability of G, we can guarantee that such a sampling oracle of size $\text{poly}\log|G|$ can be provided for every G. For H we are not aware of a similar result which allows for a presentation of its elements, and providing access to the group operation with size $\text{poly}\log|H|$. Hence we are forced to make this an explicit assumption.

Say that a group G is k-supersolvable if $1 = G_0 \triangleleft G_1 \triangleleft \cdots \triangleleft G_k = G$ where G_i/G_{i-1} is supersolvable for every i. It seems likely that our techniques extend immediately to show that for every k there exists a constant C_k such that for every $\epsilon > 0$ there are at most ϵ^{-C_k} homomorphisms that disagree with any function $f : G \to H$ on $\delta_{G,H} - \epsilon$ fraction of inputs, provided G is k-supersolvable and H is solvable. If so, an algorithmic result would also follow. We hope to report on these extensions in a fuller version of this paper. Finally, if G and H are not solvable then it appears that we have much poorer understanding of the set of homomorphisms as codes. Indeed the behavior of $\delta_{G,H}$ is no longer clean. For example, when G and H are solvable $\delta_{G,H} = 1 - 1/p$ for some prime p. But Guo [8] shows that this is no longer necessarily true if the groups are not solvable. In particular $\delta_{A_5,A_5} = 9/10$ where A_5 is the alternating group on 5 elements.

1.2 Motivation and Contributions

The study of list-decoding of homomorphisms is motivated by a few objectives. First, an abstraction of the list-decoding algorithm highlights the minimal assumptions needed to make it work. Here our work extends the understanding in terms of reducing the dependence on commutativity (and so in principle can apply to the decoding of matrix-valued functions).

A second motivation, emerging from the works of [7, 1], is to extend combinatorial analyses of list-decoding to settings beyond those where the Johnson bound is applicable. Specifically the previous works used the Johnson bound when the target group was $\mathbb{Z}_p$ for prime p and then used the group-theoretic framework to extend the analysis first to the case of cyclic groups of prime power (so $H = \mathbb{Z}_{p^k}$ for prime p and integer k) and then to the case of general abelian groups. Each one of these steps lost in the exponent. Specifically [1] gave a function $C : \mathbb{R} \to \mathbb{R}$ such that the list size grew as $(1/\epsilon)^{C(2)}$ when $H = \mathbb{Z}_{p^k}$ and $(1/\epsilon)^{C(C(2))}$ for general groups. They didn't calculate the exponents explicitly, but $C(2) \approx 105$ and $C(C(2)) \approx 3.5 \times 10^6$. Our more general abstraction ends up cleaning up their proof significantly, and even improves their exponent significantly. Specifically, we are able to apply the inductive analysis implicit in previous works directly to the solvability chain of H (rather than working with the product structure) and this allows us to merge the two steps in previous works to get a list-size bound of $(1/\epsilon)^{C(2)}$ for all supersolvable groups. Thus the abstraction and generalization improves the list-size bounds even in the abelian case. Our analysis shows that the list-decoding radius is as large as the distance. We note that there are relatively few cases of codes that are known to be list-decodable up to their minimum distance. This property is shown to be true for folded Reed-Solomon codes [10, 9], derivative/multiplicity codes [11, 12], Reed-Muller codes [6, 4], homomorphisms between abelian groups [7, 1], and codes obtained by tensor products of any of the above [5].

Finally, a potential objective would be to get new codes with better list-decodability than existing codes. Unfortunately, this hope remains unrealized in this work as well as in [7, 1].

1.3 Overview of Proof

We first prove the combinatorial bound on the list size by following the framework developed by [1], which works as follows. First, find groups $\{1\} = H_{(0)}, H_{(1)}, \ldots, H_{(m)} = H$ in such a way that any homomorphism $\phi \in \mathrm{Hom}(G, H)$ naturally induces a homomorphism $\phi^{(i)} \in \mathrm{Hom}(G, H_{(i)})$. This gives a natural notion of "extending" a homomorphism $\psi \in \mathrm{Hom}(G, H_{(i)})$: ϕ extends ψ if $\phi^{(i)} = \psi$. One then shows inductively that if $\psi \in \mathrm{Hom}(G, H_{(i)})$ has significant agreement with $f^{(i)}$, then there are not too many $\phi \in \mathrm{Hom}(G, H)$ extending ψ with significant agreement with f. In [1], H is abelian and is decomposed as $H = \mathbb{Z}_{p_1^{r_1}}^{e_1} \oplus \cdots \oplus \mathbb{Z}_{p_r^{r_m}}^{e_m}$. One

may take $H_{(i)}$ to be the direct sum of all but the last i summands. Then every $f : G \to H$ is naturally written as $f = (f_1, \ldots, f_m)$ where $f_i : G \to \mathbb{Z}_{p_i^{r_i}}^{e_i}$, and thus $f^{(i)} = (f_1, \ldots, f_{m-i})$. Now, to show the inductive claim for H, they reduce to the special cases where $H = \mathbb{Z}_p^r$ and where $H = \mathbb{Z}_{p^r}$, and go through the same approach for the special cases too. This goes through the "special intersecting family" theorem of [1] twice, resulting in a huge blowup in the exponent of the list size. Our proof differs from that of [1] as we prove the full inductive claim directly, without reducing to any special cases, resulting in a much smaller exponent. However, for technical reasons, we only manage to use this approach when the smallest prime divisor of $|G|$ also divides $|H|$. In the general case, we reduce to the previous case by decomposing G as a semidirect product.

The algorithmic results are a straightforward generalization of those of [1]. In particular, one merely needs to find the correct way to generalize the algorithms (replacing the direct product presentation of G with a polycyclic presentation) and verifying that the same analysis goes through.

2 Preliminaries

2.1 Group Homomorphisms

Let G and H be finite groups, with homomorphisms $\mathrm{Hom}(G, H)$. A function $\phi : G \to H$ is a (left) affine homomorphism if there exists $h \in H$ and $\phi_0 \in \mathrm{Hom}(G, H)$ such that $\phi(g) = h\phi_0(g)$ for every $g \in G$. We use $\mathrm{aHom}(G, H)$ to denote the set of left affine homomorphisms from G to H. Note that the set of left affine homomorphisms equals the set of right affine homomorphisms, since

$$h\phi_0(g) = (h\phi_0(g)h^{-1})h$$

and $\psi_0(g) \triangleq h\phi_0(g)h^{-1}$ is a homomorphism.

The *equalizer* of two functions $f, g : G \to H$, denoted $\mathrm{Eq}(f, g)$, is the subset of G on which f and g agree, i.e.

$$\mathrm{Eq}(f, g) \triangleq \{x \in G \mid f(x) = g(x)\}.$$

More generally, if $\Phi \subseteq \{f : G \to H\}$ is a collection of functions, then the *equalizer* of Φ is the set

$$\mathrm{Eq}(\Phi) \triangleq \{x \in G \mid f(x) = g(x) \ \ \forall f, g \in \Phi\}.$$

In the theory of error correcting codes, the usual measure of distance between two strings is the relative Hamming distance, which is the fraction of symbols on which they differ. In the context of group homomorphisms, we find it more convenient to study the complementary notion, the fractional agreement. We define the *agreement* $\mathrm{agr}(f, g)$ between two functions $f, g : G \to H$ to be the quantity

$$\mathrm{agr}(f, g) \triangleq \frac{|\mathrm{Eq}(f, g)|}{|G|}.$$

The *maximum agreement* of the code $\mathrm{aHom}(G, H)$, denoted by $\Lambda_{G,H}$, is defined as

$$\Lambda_{G,H} \triangleq \max_{\substack{\phi,\psi \in \mathrm{aHom}(G,H) \\ \phi \neq \psi}} \mathrm{agr}(\phi, \psi)$$

Recall that a group H is said to be *nilpotent* if it has a series of subgroups $\{1_G\} = G_0 \triangleleft G_1 \triangleleft \cdots \triangleleft G_k = G$ such that for each i the commutator subgroup $[G, G_i]$, generated by all $g^{-1}h^{-1}gh$ for $g \in G$ and $h \in G_i$, is a subgroup of G_{i-1}. such that for each i, the commutator subgroup $[G, G_i]$ is a subgroup of G_{i-1}. The following theorem gives the value of $\Lambda_{G,H}$ when G is solvable or H is nilpotent.

▶ **Theorem 2.1** ([8]). *Suppose G and H are finite groups and G is solvable or H is nilpotent. Then*

$$\Lambda_{G,H} = \frac{1}{p}$$

where p is the smallest prime divisor of $\gcd(|G|, |H|)$ such that G has a normal subgroup of index p. If no such p exists, then $|\operatorname{Hom}(G, H)| = 1$; in particular, $\Lambda_{G,H} = 0$.

We also need the following proposition relating $\Lambda_{G.H}$ and $\Lambda_{N,H}$ when $N \triangleleft G$ and G can be written as a semidirect product of N with some other group G_1. (Recall that the semidirect product of two groups A and B, denoted $A \rtimes B$, is defined when elements of B act on the elements of A. The elements of $A \rtimes B$ are pairs (a, b) with $a \in A$ and $b \in B$ and $(a, b) \cdot (c, d) = ((a \cdot c^b, b \cdot d).)$

▶ **Proposition 2.2.** *If G and H are finite groups and $G = N \rtimes G_1$ for some normal subgroup $N \triangleleft G$ and subgroup $G_1 \leq G$ and $|\operatorname{Hom}(G_1, H)| = 1$, then every $\phi \in \operatorname{aHom}(G, H)$ is of the form $\phi(xy) = \psi(x)$ for some $\psi \in \operatorname{aHom}(N, H)$ and every $x \in N$ and $y \in G_1$. In particular,*

$$\Lambda_{G,H} \leq \Lambda_{N,H}\,.$$

2.2 Some Facts About Supersolvable Groups

▶ **Proposition 2.3.** *If G is a finite supersolvable group and $|G| = p_1 \cdots p_k$, where $p_1 \geq \cdots \geq p_k$ are primes, then G has a normal cyclic series*

$$\{1_G\} = G_0 \triangleleft G_1 \triangleleft \cdots \triangleleft G_k = G$$

where each $G_i/G_{i-1} \cong \mathbb{Z}_{p_i}$.

The following proposition allows us to decompose a finite supersolvable group as a semidirect product whose components have coprime orders.

▶ **Proposition 2.4.** *Let G be a finite supersolvable group and $|G| = p_1^{r_1} \cdots p_m^{r_m}$, where $p_1 > \cdots > p_m$ are prime. For any $k \in [m]$, G has a normal subgroup $N_k \triangleleft G$ such that $|N_k| = p_1^{r_1} \cdots p_k^{r_k}$, $|G/N_k| = p_{k+1}^{r_{k+1}} \cdots p_m^{r_m}$, and $G = N_k \rtimes G/N_k$.*

2.3 Special Intersecting Families

▶ **Definition 2.5** (Special intersecting family)**.** Fix an ambient set X. For any subset $S \subseteq X$, define the *density of S in X* to be

$$\mu(S) = \frac{|S|}{|X|}.$$

A collection $S_1, \ldots, S_\ell \subseteq X$ of subsets is a (ρ, τ, c)-*special intersecting family* if the following hold:

1. $\mu(S_i) \geq \rho$ for each i;
2. $\mu(S_i \cap S_j) \leq \rho$ whenever $i \neq j$;
3. $\sum_{i=1}^{\ell} (\mu(S_i) - \rho)^c \leq 1$;
4. If $J \subseteq I \subseteq [\ell]$, $|J| \geq 2$, and $\mu(S_I) > \tau$, then $S_I = S_J$, where $S_K = \cap_{i \in K} S_i$ for any $K \subseteq [\ell]$;

For our bounds on the combinatorial list-decodability, we use the same outline as that of [1]. In particular, this involves analyzing the agreement sets of homomorphisms with the given function and showing that they form a special intersecting family. The following result of [1] allows us to deduce bounds on the sizes of the agreement sets in terms of the size of the union.

▶ **Theorem 2.6** ([1, Theorem 3.2]). *For every $c < \infty$, there exists $C = C(c) < \infty$ such that the following holds: if $S_1, \ldots, S_\ell$ form a (ρ, ρ^2, c)-special intersecting family, with $\mu(S_i) = \rho + \alpha_i$ and $\mu(\cup_i S_i) = \rho + \alpha$, then*

$$\alpha^C \geq \sum_{i=1}^{\ell} \alpha_i^C. \tag{1}$$

In fact, one can take $C(c) = 2c \cdot (c+1)(4 + (c+1)\log_2 3)$.

We refer to $C(c)$ as the *special intersecting number for c*.

We will also use the following q-ary Johnson bound (see the appendix of [1] for a proof).

▶ **Proposition 2.7** (q-ary Johnson Bound). *Let $f, \phi_1, \ldots, \phi_\ell : [n] \to [q]$ be functions satisfying the following properties:*

1. $\mathrm{agr}(f, \phi_i) = \frac{1}{q} + \alpha_i$ *for* $\alpha_i \geq 0$
2. $\mathrm{agr}(\phi_i, \phi_j) \leq \frac{1}{q}$ *for every* $i \neq j$.

Then $\sum_{i=1}^{\ell} \alpha_i^2 \leq 1$.

3 List-decoding Radius for Supersolvable Groups

3.1 Preliminary Notation and Definitions

If H is supersolvable, we may write

$$H = H_0 \triangleright H_1 \triangleright \cdots \triangleright H_m = \{1\}$$

where $H_{i-1}/H_i \cong \mathbb{Z}_{p_i}$. For $k \in [m]$, define $H_{(k)} \triangleq H/H_k$, which is a group since H_k is normal in H. In particular, $H_{(0)} = \{1\}$ and $H_{(m)} = H$.

Given $f : G \to H$ and $k \in [m]$, define $f^{(k)} : G \to H_{(k)}$ and $f^{(-k)} : G \to H_k$ as follows. Define $f^{(k)} : G \to H_{(k)}$ to be f composed with the natural quotient map, sending $x \in G$ to the coset $f(x)H_k$ of H_k. Therefore, $f^{(k)}$ is an (affine) homomorphism if f is. To define the latter map, we need to choose, for each $i \in [0, m-1]$, an element $y_i \in H_i \setminus H_{i+1}$. Then each k-tuple $(a_0, \ldots, a_{k-1})$, where $0 \leq a_j \leq p-1$, corresponds to a distinct coset $y_0^{a_0} \cdots y_{k-1}^{a_{k-1}} H_k$. If $f(x)H_k = y_0^{a_0} \cdots y_{k-1}^{a_{k-1}} H_k$, then define $f^{(-k)}(x) \triangleq y_{k-1}^{-a_{k-1}} \cdots y_0^{-a_0} f(x)$. Note that $f^{(-k)}(x) \in H_k$ but $f^{(-k)}$ may not be a homomorphism in general (even if f is). Also, note that f is determined by $f^{(k)}$ and $f^{(-k)}$: if $f^{(k)}(x) = y_0^{a_0} \cdots y_{k-1}^{a_{k-1}} H_k$, then $f(x) = y_0^{a_0} \cdots y_{k-1}^{a_{k-1}} f^{(-k)}(x)$.

If $i < j$ and $\phi : G \to H_{(i)}$ and $\psi : G \to H_{(j)}$, then ψ *extends* ϕ if $\psi^{(i)} = \phi$. Here, $\psi^{(i)}$ makes sense, because $H_j < H_i$, and so we get a chain $H_0/H_j \triangleright H_1/H_j \triangleright \cdots \triangleright H_j/H_j = \{1\}$ induced by the original chain for H, and so $\psi^{(i)}$ is just ψ composed by modding out by H_i/H_j. One can then define $\psi^{(-i)}$ to make sense too.

3.2 Combinatorial Bounds for Agreement $\Lambda_{G,H} + \epsilon$

We begin with the case where the smallest prime divisor of $|G|$ also divides $|H|$.

▶ **Theorem 3.1.** *There exists a universal constant $C < \infty$ such that whenever G and H are finite supersolvable groups and the smallest prime divisor p of $|G|$ also divides $|H|$, then for any $f : G \to H$ and $\epsilon > 0$, there are at most $(1/\epsilon)^C$ affine homomorphisms $\phi \in \mathrm{aHom}(G, H)$ such that $\mathrm{agr}(\phi, f) \geq \frac{1}{p} + \epsilon$.*

Proof. We prove the theorem for $C = C(2)$ where $C(c)$ denotes the special intersecting number of c, as given by Theorem 2.6. Henceforth let $C = C(2)$. Let $p_1 \leq \cdots \leq p_m$ be primes such that $|H| = p_1 \cdots p_m$. By Proposition 2.3, H has a normal cyclic series

$$H = H_0 \triangleright H_1 \triangleright \cdots \triangleright H_m = \{1_H\}$$

where $H_{i-1}/H_i \cong \mathbb{Z}_{p_i}$ for each i.

▶ Claim 3.2. For $k \in [0, m]$, if $\phi \in \mathrm{aHom}(G, H_{(k)})$ satisfies $\mathrm{agr}(\phi, f^{(k)}) = \frac{1}{p} + \alpha$ for some $\alpha \geq \epsilon$, then the number of $\psi \in \mathrm{aHom}(G, H)$ extending ϕ with $\mathrm{agr}(\psi, f) \geq \frac{1}{p} + \epsilon$ is at most $(\alpha/\epsilon)^C$.

Proof. We induct backwards on k. The base case $k = m$ is trivial. Now suppose $k < m$ and the claim holds for $k + 1$. Let $\phi_1, \ldots, \phi_\ell \in \mathrm{aHom}(G, H_{(k+1)})$ be all the homomorphisms extending ϕ with $\mathrm{agr}(\phi_i, f^{(k+1)}) \geq \frac{1}{p} + \epsilon$. Define $\alpha_i \triangleq \mathrm{agr}(\phi_i, f^{(k+1)}) - \frac{1}{p}$. Define $S_i \triangleq \mathrm{Eq}(\phi_i, f^{(k+1)})$. We claim that $S_1, \ldots, S_\ell$ form a $\left(\frac{1}{p}, \frac{1}{p^2}, 2\right)$-special intersecting family. Before we prove this, we show how it implies the claim. By Theorem 2.6, $(\alpha')^C \geq \sum_{i=1}^{\ell} \alpha_i^C$, where $\alpha' = \mu(\cup_i S_i) - \frac{1}{p}$. But $\cup_i S_i \subseteq \mathrm{Eq}(\phi, f)$, so $\alpha \geq \alpha'$, and thus $\alpha^C \geq \sum_{i=1}^{\ell} \alpha_i^C$. Moreover, every $\psi \in \mathrm{aHom}(G, H)$ extending ϕ with $\mathrm{agr}(\psi, f) \geq \frac{1}{p} + \epsilon$ must extend one of the ϕ_i. By induction, there are at most $(\alpha_i/\epsilon)^C$ such ψ extending ϕ_i. Hence, there are at most $\sum_{i=1}^{\ell} (\alpha_i/\epsilon)^C \leq (\alpha/\epsilon)^C$ such ψ extending ϕ.

Now, we show that $S_1, \ldots, S_\ell$ form a $\left(\frac{1}{p}, \frac{1}{p^2}, 2\right)$-special intersecting family. We verify the four properties:

1. By definition, we have $\mu(S_i) = \frac{1}{p} + \alpha_i \geq \frac{1}{p}$.
2. If $i \neq j$, then since $\phi_i, \phi_j \in \mathrm{aHom}(G, H_{(k+1)})$, we have $S_i \cap S_j \subseteq \mathrm{Eq}(\phi_i, \phi_j)$ and therefore $\mu(S_i \cap S_j) \leq \mathrm{agr}(\phi_i, \phi_j) \leq \Lambda_{G, H_{(k+1)}} \leq \Lambda_{G,H} \leq \frac{1}{p}$.
3. Define $g \triangleq (f^{(k+1)})^{(-k)} : G \to H_k/H_{k+1} \cong \mathbb{Z}_{p_{k+1}}$ and define $\psi_i \triangleq \phi^{(-k)} : G \to H_k/H_{k+1} \cong \mathbb{Z}_{p_{k+1}}$. If $\phi_i(x) = f^{(k+1)}(x)$, then $\psi_i(x) = g(x)$, so certainly $\mathrm{agr}(g, \psi_i) \geq \mathrm{agr}(f^{(k+1)}, \phi_i) = \frac{1}{p} + \alpha_i$. Moreover, if $i \neq j$, since ϕ_i, ϕ_j both extend ϕ, then $\phi_i(x) = \phi_j(x)$ if and only if $\psi_i(x) = \psi_j(x)$, so $\mathrm{agr}(\psi_i, \psi_j) = \mathrm{agr}(\phi_i, \phi_j) \leq \Lambda_{G, H_{(k+1)}} \leq \Lambda_{G,H} = \frac{1}{p}$.
4. Suppose $J \subseteq I$, $|J| \geq 2$, and $\mu(S_I) > 1/p^2$. Define $\Phi_I \triangleq \{\phi_i \mid i \in I\}$ and define Φ_J similarly. Then $S_I \subseteq \mathrm{Eq}(\Phi_I)$ and $S_J \subseteq \mathrm{Eq}(\Phi_J)$, and since $|J| \geq 2$, we have $1/p^2 < \mu(\mathrm{Eq}(\Phi_I)) \leq \mu(\mathrm{Eq}(\Phi_J)) \leq 1/p$. But $\mu(\mathrm{Eq}(\Phi_J))/\mu(\mathrm{Eq}(\Phi_I))$ divides $|G|$ and p is the smallest prime divisor of $|G|$, so it must be that $\mu(\mathrm{Eq}(\Phi_I)) = \mu(\mathrm{Eq}(\Phi_J))$, and hence $\mathrm{Eq}(\Phi_I) = \mathrm{Eq}(\Phi_J)$. Fix any $j \in J$. Then $S_I = S_j \cap \mathrm{Eq}(\Phi_I) = S_j \cap \mathrm{Eq}(\Phi_J) = S_J$. ◀

The theorem follows by taking $k = 0$ in the claim. ◀

Before we prove the general case, we first prove a useful lemma. In what follows, for any code $\mathcal{C} \subseteq \Sigma^n$ and agreement parameter $a \in [0, 1]$, define $\ell(\mathcal{C}, a)$ to be the quantity

$$\ell(\mathcal{C}, a) \triangleq \max_{w \in \Sigma^n} |\{c \in \mathcal{C} \mid \mathrm{agr}(c, w) \geq a\}|.$$

▶ **Lemma 3.3.** *Let $\mathcal{C} \subseteq \Sigma^n$ be a code. If $s > r \geq 1$, and $\mathcal{C}_r \triangleq \{(\underbrace{c,\ldots,c}_{r}) \in \Sigma^{rn} \mid c \in \mathcal{C}\}$ and $\mathcal{C}_s \triangleq \{(\underbrace{(c,\ldots,c)}_{s}) \in \Sigma^{sn} \mid c \in \mathcal{C}\}$, then for any $a \in [0,1]$,*

$$\ell(\mathcal{C}_r, a) \leq \ell(\mathcal{C}_s, \lfloor s/r \rfloor (r/s) \cdot a).$$

Proof. Let $w \in \Sigma^{rn}$ such that $|\{\underbrace{(c,\ldots,c)}_{r} \in \mathcal{C}_r \mid \mathrm{agr}(\underbrace{(c,\ldots,c)}_{r}, w) \geq a\}| = \ell(\mathcal{C}_r, a)$. Define $w' \in \Sigma^{sn}$ by $w' = (\underbrace{w,\ldots,w}_{\lfloor s/r \rfloor}, w'')$, where $w'' \in \Sigma^{(s-\lfloor s/r \rfloor r)n}$ is defined arbitrarily. Then for each $c \in \mathcal{C}$ such that $\mathrm{agr}(\underbrace{(c,\ldots,c)}_{r}, w) \geq a$,

$$\begin{aligned} \mathrm{agr}(\underbrace{(c,\ldots,c)}_{s}, w') &\geq \frac{1}{sn}\left(\lfloor s/r \rfloor \cdot rn \cdot \mathrm{agr}(\underbrace{(c,\ldots,c)}_{r}, w)\right) \\ &\geq \left\lfloor \frac{s}{r} \right\rfloor \frac{r}{s} \cdot a. \end{aligned}$$

◀

▶ **Theorem 3.4.** *There exists a universal constant $C < \infty$ such that whenever G and H are finite supersolvable groups, then for any $f : G \to H$ and $\epsilon > 0$, there are at most $(1/\epsilon)^C$ affine homomorphisms $\phi \in \mathrm{aHom}(G,H)$ such that $\mathrm{agr}(\phi, f) \geq \Lambda_{G,H} + \epsilon$.*

Proof. We prove the theorem for $C = C(2)$, in particular using the fact that Theorem 2.6 holds for this constant. Let p be the smallest prime divisor of $\gcd(|G|,|H|)$ such that G has a normal subgroup of index p, so that $\Lambda_{G,H} = \frac{1}{p}$ (Theorem 2.1). If p is the smallest prime divisor of $|G|$, then the result follows from Theorem 3.1, so suppose p is not the smallest prime divisor of $|G|$. By Proposition 2.4, we can write $G = N \rtimes G'$ for some proper normal subgroup $N \triangleleft G$ where p is the smallest prime divisor of $|N|$ and every prime dividing $|G'|$ is smaller than p, and therefore $\gcd(|G'|,|H|) = 1$. By Proposition 2.2, every $\phi \in \mathrm{aHom}(G,H)$ is of the form $\phi(x,y) = \psi(x)$ for $x \in N$ and $y \in G'$. Thus, $\mathrm{aHom}(G,H)$ is isomorphic to the code

$$\mathcal{C}_r \triangleq \{(\underbrace{\psi,\ldots,\psi}_{r}) \mid \psi \in \mathcal{C}\}$$

where $\mathcal{C} = \mathrm{aHom}(N,H)$ and $r = |G'|$. Let $q > \max\{|G|,|H|\}$ be a prime and consider the group $G'' \triangleq N \oplus \mathbb{Z}_q$, which is supersolvable. Then $\mathrm{aHom}(G'',H)$ is isomorphic to the code

$$\mathcal{C}_q \triangleq \{(\underbrace{\psi,\ldots,\psi}_{q}) \mid \psi \in \mathcal{C}\}.$$

Letting $a \triangleq \frac{1}{p} + \epsilon$, applying Lemma 3.3 and Theorem 3.1 (using the fact that the smallest prime divisor of $|G''|$ also divides $|H|$), we get an upper bound of

$$\left(\frac{1}{(\lfloor q/|G'| \rfloor (|G'|/q) - 1)\frac{1}{p} + \lfloor q/|G'| \rfloor (|G'|/q) \cdot \epsilon}\right)^C \leq \left(\frac{1}{\left(1 - \frac{|G'|}{q}\right)\epsilon - \frac{|G'|}{q}\frac{1}{p}}\right)^C$$

affine homomorphisms $\phi \in \mathrm{aHom}(G,H)$ with $\mathrm{agr}(\phi, f) \geq \frac{1}{p} + \epsilon$. By taking $q \to \infty$, the above upper bound approaches $(1/\epsilon)^C$. ◀

3.3 Exponential List Size for Agreement $\Lambda_{G,H}$

We conclude this section by showing that if G is solvable, then the list size for agreement $\Lambda_{G,H}$ can be exponential in $\log|G| + \log|H|$, showing that the list-decoding distance we achieve is optimal. In other words, we have identified the list-decoding radius for $\mathrm{aHom}(G,H)$ when G and H are supersolvable.

In fact, we observe that the list size can be $\Omega(|G| \cdot |H|)$ even just for abelian G and H, when $\Lambda_{G,H} = \frac{1}{p}$ is fixed. Let $G = \mathbb{Z}_p^n$ and $H = \mathbb{Z}_p^m$, so that $\Lambda_{G,H} = \frac{1}{p}$. Consider the maps $\phi_{a,b}$, where $a \in \mathbb{Z}_p^n$ and $b \in \mathbb{Z}_p^m$ are nonzero vectors, defined by

$$\phi_{a,b}(x_1, \ldots, x_n) = (a_1x_1 + \cdots + a_nx_n)b.$$

Note that $\mathrm{agr}(\phi_{a,b}, 0) = \frac{1}{p}$. Moreover, there are $p^n - 1$ choices for a and $p^m - 1$ choices for b, and $\phi_{a,b} = \phi_{c,d}$ if and only if there exists $\lambda \in \mathbb{Z}_p^*$ such that $c = \lambda a$ and $b = \lambda d$. So the number of distinct homomorphisms agreeing with the zero function is $\frac{(p^n-1)(p^m-1)}{p-1} = \Omega(|G| \cdot |H|) = \exp(\log|G| + \log|H|)$.

4 Algorithm for Supersolvable G

In this section we give a local list-decoding algorithm for the set of homomorphisms from G to H whenever G and H are supersolvable.

▶ **Definition 4.1.** A probabilistic oracle algorithm $\mathcal{A}$ for list decoding homomorphisms takes as input two groups G and H and has oracle access to a function $f : G \to H$. The algorithm $\mathcal{A}$ is a (λ, T)-local list decoder for $\mathrm{aHom}(G,H)$ if, for every function $f : G \to H$, $\mathcal{A}^f$ runs in time T and outputs a list $L \subset \mathrm{aHom}(G,H)$ such that with probability at least $3/4$, it holds that if $\phi \in \mathrm{aHom}(G,H)$ and $\mathrm{agr}(f,\phi) \geq \lambda$, then $\phi \in L$.

▶ **Theorem 4.2.** *There exists an algorithm $\mathcal{A}$ such that for every pair of finite groups G, H where G is solvable and H is supersolvable, and every $\epsilon > 0$, $\mathcal{A}$ is a $(\Lambda_{G,H} + \epsilon, \mathrm{poly}(\log|G|, \log|H|, \frac{1}{\epsilon}))$-local list decoder for $\mathrm{aHom}(G,H)$, provided that $\mathcal{A}$ has oracle access to the multiplication table of H and $\mathrm{aHom}(G,H)$ has a list-size bound of $(1/\epsilon)^{O(1)}$.*

4.1 Algorithm

Let

$$G = G_k \triangleright G_{k-1} \triangleright \cdots \triangleright G_0 = \{1_G\}$$

be a subnormal cyclic series, with $G_i/G_{i-1} \cong \mathbb{Z}_{p_i}$, $p_1 \geq p_2 \geq \cdots \geq p_k$ and representatives $g_i \in G_i \setminus G_{i-1}$. Our main algorithm is Algorithm 1, which uses Algorithms 2 and 3 as subroutines.

The analysis is the same as in [1].

Algorithm 1 List decode

procedure ListDecode(f,G,H)
  $\mathcal{L} \leftarrow \emptyset$
  repeat
    $S_0 \leftarrow \emptyset$
    for $i = 1$ to k **do**
      $S'_i \leftarrow \text{Extend}(i, S_{i-1})$
      $S_i \leftarrow \text{Prune}(i, S'_i)$
    end for
    for all $\phi \in S_k$ **do**
      $B \leftarrow \text{FrequentValues}(x \mapsto f(x)\phi(x)^{-1}, \Lambda_{G,H} + \epsilon/2)$
      $\mathcal{L} \leftarrow \mathcal{L} \cup \{x \mapsto b\phi(x) \mid b \in B\}$
    end for
  until $C \log \frac{1}{\epsilon}$ times
end procedure

Algorithm 2 Extend

procedure Extend(i,S)
  $S' \leftarrow \emptyset$
  for all $\phi \in S$ **do**
    repeat
      Pick $(\alpha_{i+1}, \ldots, \alpha_k) \in \mathbb{Z}_{p_{i+1}} \oplus \cdots \oplus \mathbb{Z}_{p_k}$ uniformly at random
      $s \leftarrow g_k^{\alpha_k} \cdots g_{i+1}^{\alpha_{i+1}}$
      Pick $y_1, y_2 \in G_{i-1}$ and $c_1, c_2 \in \mathbb{Z}_{p_i}$ uniformly at random
      if $c_1 - c_2$ is invertible modulo $p_1 \cdots p_i$ **then**
        $\gamma \leftarrow (c_1 - c_2)^{-1} \in \mathbb{Z}^{\oplus} p_1 \cdots p_i$
        $a \leftarrow (\phi(y_2) f(s g_i^{c_2} y_2)^{-1} f(s g_i^{c_1} y_1) \phi(y_1)^{-1})^{\gamma}$
        Define $\theta : G_i \to H$ by $\theta(g_i^c x) = a^c \phi(x)$
        $S' \leftarrow S' \cup \{\theta\}$
      end if
    until $(\log |G| \log |H| \frac{1}{\epsilon})^4$ times
  end for
  return S'
end procedure

Algorithm 3 Prune

procedure Prune(i,S)
  $S' \leftarrow \emptyset$
  repeat
    Pick $(\alpha_{i+1}, \ldots, \alpha_k) \in \mathbb{Z}_{p_{i+1}} \oplus \cdots \oplus \mathbb{Z}_{p_k}$ uniformly at random
    $s \leftarrow g_k^{\alpha_k} \cdots g_{i+1}^{\alpha_{i+1}}$
    for all $\phi \in S$ **do**
      $B \leftarrow \text{FrequentValues}(x \mapsto f(sx)\phi(sx)^{-1}, \Lambda_{G,H} + \epsilon/2)$
      if $|B| \geq 1$ **then**
        $S' \leftarrow S' \cup \{\phi\}$
      end if
    end for
  until $(\log |G| \log |H| \frac{1}{\epsilon})^2$ times
  if $|S'| > (\log |G| \log |H| \frac{1}{\epsilon})^{2C}$ **then**
    return error
  end if
  return S'
end procedure

References

1. Irit Dinur, Elena Grigorescu, Swastik Kopparty, and Madhu Sudan. Decodability of group homomorphisms beyond the Johnson bound. In Dwork [2], pages 275–284.
2. Cynthia Dwork, editor. *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*. ACM, 2008.
3. Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In David S. Johnson, editor, *STOC*, pages 25–32. ACM, 1989.
4. Parikshit Gopalan. A Fourier-analytic approach to Reed-Muller decoding. *IEEE Transactions on Information Theory*, 59(11):7747–7760, 2013.
5. Parikshit Gopalan, Venkatesan Guruswami, and Prasad Raghavendra. List decoding tensor products and interleaved codes. *SIAM Journal on Computing*, 40(5):1432–1462, 2011.
6. Parikshit Gopalan, Adam R. Klivans, and David Zuckerman. List-decoding Reed-Muller codes over small fields. In Dwork [2], pages 265–274.
7. Elena Grigorescu, Swastik Kopparty, and Madhu Sudan. Local decoding and testing for homomorphisms. In Josep Díaz, Klaus Jansen, José D. P. Rolim, and Uri Zwick, editors, *APPROX-RANDOM*, volume 4110 of *Lecture Notes in Computer Science*, pages 375–385. Springer, 2006.
8. Alan Guo. Group homomorphisms as error correcting codes, April 2014. `http://arxiv.org/abs/1404.3447`.
9. Venkatesan Guruswami. Linear-algebraic list decoding of folded Reed-Solomon codes. In *IEEE Conference on Computational Complexity*, pages 77–85. IEEE Computer Society, 2011.
10. Venkatesan Guruswami and Atri Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.
11. Venkatesan Guruswami and Carol Wang. Optimal rate list decoding via derivative codes. In Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim, editors, *APPROX-RANDOM*, volume 6845 of *Lecture Notes in Computer Science*, pages 593–604. Springer, 2011.
12. Swastik Kopparty. List-decoding multiplicity codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:44, 2012.

Evading Subspaces Over Large Fields and Explicit List-decodable Rank-metric Codes*

Venkatesan Guruswami and Carol Wang

Computer Science Department, Carnegie Mellon University
Pittsburgh, PA
{venkatg,wangc}@cs.cmu.edu

Abstract

We construct an *explicit* family of *linear* rank-metric codes over any field $\mathbb{F}_h$ that enables efficient list decoding up to a fraction ρ of errors in the rank metric with a rate of $1-\rho-\varepsilon$, for any desired $\rho \in (0,1)$ and $\varepsilon > 0$. Previously, a Monte Carlo construction of such codes was known, but this is in fact the first explicit construction of positive rate rank-metric codes for list decoding beyond the unique decoding radius.

Our codes are explicit subcodes of the well-known Gabidulin codes, which encode linearized polynomials of low degree via their values at a collection of linearly independent points. The subcode is picked by restricting the message polynomials to an $\mathbb{F}_h$-subspace that evades certain structured subspaces over an extension field $\mathbb{F}_{h^t}$. These structured spaces arise from the linear-algebraic list decoder for Gabidulin codes due to Guruswami and Xing (STOC'13). Our construction is obtained by combining subspace designs constructed by Guruswami and Kopparty (FOCS'13) with subspace-evasive varieties due to Dvir and Lovett (STOC'12).

We establish a similar result for subspace codes, which are a collection of subspaces, every pair of which have low-dimensional intersection, and which have received much attention recently in the context of network coding. We also give explicit subcodes of folded Reed-Solomon (RS) codes with small folding order that are list-decodable (in the Hamming metric) with optimal redundancy, motivated by the fact that list decoding RS codes reduces to list decoding such folded RS codes. However, as we only list decode a *subcode* of these codes, the Johnson radius continues to be the best known error fraction for list decoding RS codes.

1998 ACM Subject Classification E.4 Coding and Information Theory

Keywords and phrases list-decoding, pseudorandomness, algebraic coding, explicit constructions

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.748

1 Introduction

This paper considers the problem of constructing explicit list-decodable rank-metric codes. A *rank-metric code* is a collection of matrices $M \in \mathbb{F}_h^{n\times t}$ over a finite field $\mathbb{F}_h$ for fixed n, t. The rate of a rank-metric code is $\log_h|\mathcal{C}|/(nt)$, and the distance measure between two codewords is the rank over $\mathbb{F}_h$ of their difference; that is, $\mathrm{dist}(M_1, M_2) = \mathrm{rank}_{\mathbb{F}_h}(M_1 - M_2)$. We will be interested in *linear* rank-metric codes, where $\mathcal{C}$ is a subspace over $\mathbb{F}_h$.

Rank-metric codes have found applications in network coding [23] and public-key cryptography [8, 17], among other areas. They can also be thought of as space-time codes over finite fields, and conversely can be used to construct space-time codes, eg. in [19, 18]. Unique decoding algorithms for rank-metric codes were shown in [5] to be closely related to

* Research supported in part by NSF CCF-0963975.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 748–761

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

the so-called Low-rank Recovery problem, in which the task is to recover a matrix M from few inner products $\langle M, H\rangle$. The authors of [5] use their low-rank recovery techniques to construct rank-metric codes over any field, and show that they can be efficiently decoded.

In this work, we will consider subcodes of *Gabidulin codes*, which are analogues of Reed-Solomon codes for the rank-metric. A Gabidulin code (denoted $\mathcal{C}_G(h; n, t, k)$) encodes h-linearized polynomials over $\mathbb{F}_{h^t}$ of h-degree less than k by $\big(f(\alpha_1), \ldots, f(\alpha_n)\big)^T$, where the $\alpha_i \in \mathbb{F}_{h^t}$ are linearly independent over $\mathbb{F}_h$, and $f(\alpha_j)$ is thought of as a column vector in $\mathbb{F}_h^t$ under a fixed basis of $\mathbb{F}_{h^t}$ over $\mathbb{F}_h$. This is a rank-metric code of rate k/n and minimum distance $n - k + 1$.

We say that a rank-metric code $\mathcal{C}$ can be decoded from up to e *rank errors* if any codeword $M \in \mathcal{C}$ can be recovered from $M + E$ whenever $E \in \mathbb{F}_h^{n\times t}$ has rank at most e. Gabidulin codes can be uniquely decoded from $(n-k)/2$ rank errors by adapting algorithms for Reed-Solomon decoding, as in [6, 7, 22], among others, but it is still open whether they can be *list-decoded* from a larger fraction of errors. We recall that in the list-decoding problem the decoder must output all codewords within the stipulated radius from the noisy codeword it is given as input. It is known that Gabidulin codes *cannot* be list-decoded with a polynomial list size from an error fraction exceeding $1 - \sqrt{R}$ [4, 24]. However, as we show in this work, we can explicitly pick a good subcode of the Gabidulin code, with only a minor loss in rate, that enables efficient list-decoding all the way up to a fraction $(1 - R)$ of errors.

The primary difficulty in previous work on list-decoding Gabidulin codes has been the fact that in contrast to Reed-Solomon codes, where the field size grows with the dimension of the code, for Gabidulin codes, the *dimension* of the ambient space grows with the dimension of the code. This forces us to work over fields whose size can be exponential in the code dimension.

To address this, we show how to find linear list-decodable subcodes of certain Gabidulin codes by adapting the subspace designs of [9] for use over large fields. The key observation, first made in [14], is that although applying a linear-algebraic list-decoder gives a subspace over a field which is too large, the subspace has additional structure which can then be "evaded" using *pseudorandom* subcodes, yielding a polynomial list size.

We combine recent constructions of *subspace designs* [9] and *subspace-evasive sets* [1] in order to give an explicit construction of a subcode (in fact, subspace) of the Gabidulin code which has small intersection with the output of the linear-algebraic list-decoder of [14]. In particular, we show (Theorem 12):

▶ **Theorem** (Main). *For every field $\mathbb{F}_h$, $\varepsilon > 0$ and integer $s > 0$, there exists an explicit $\mathbb{F}_h$-linear subcode of the Gabidulin code $\mathcal{C}_G(h; n, t, k)$ with evaluation points $\alpha_1, \ldots, \alpha_n$ spanning a subfield $\mathbb{F}_{h^n}$ that has (i) rate $(1 - 2\varepsilon)k/n$, and (ii) is list-decodable from $s(n - k)/(s + 1)$ rank errors. The final list is contained in an $\mathbb{F}_h$-subspace of dimension $O(s^2/\varepsilon^2)$.*

Note that the fraction of errors corrected approaches the information-theoretic limit of $(1 - R)$ (where $R = k/n$ is the rate) as the parameter s grows. The authors of [14] give a *Monte Carlo* construction of a subcode of the same Gabidulin code satisfying these guarantees, in fact with a better list size of $O(1/\varepsilon)$. We give an *explicit* subcode, with a worse guarantee on the list size (which, however, is still bounded by a constant depending only on ε).

We also note that the above theorem gives the *first explicit construction* of positive rate rank-metric codes even for list-decoding from a number of errors which is more than half the distance (and in particular for list decoding beyond a fraction $(1-R)/2$ of errors). Previous explicit codes only achieved polynomially small rate [10].

Our techniques also imply analogous results for *subspace codes*, which can be thought of as a basis-independent form of rank-metric codes. They were defined in [16] to address the problem of non-coherent linear network coding in the presence of errors, and have received much attention lately ([2, 20, 3], etc). The authors of [16] also define the Kötter-Kschischang (KK) codes, which, like Gabidulin codes, are linearized variants of Reed-Solomon codes. List-decoding of a folded variant of the KK code was considered in [10] and [21]. However, both of these papers could only guarantee a polynomial list size when the rate of the code was polynomially small, and the question of constructing constant rate list-decodable subspace codes remained open. Note that [14] was able, similarly to the case of rank-metric codes, to give a Monte Carlo construction of a constant rate list-decodable subcode.

In this work, we give the first explicit construction of high-rate subspace codes which are list-decodable past the unique decoding radius (stated in Theorem 20). Our construction does not use folding, but instead takes subcodes of certain KK codes.

Additionally, we use our ideas to list-decode a subcode of the folded Reed-Solomon code where the folding parameter is of low order (see Corollary 16 for a formal statement). List-decoding of the folded Reed-Solomon code up to list-decoding capacity where the folding parameter is primitive was first shown in [11]. In [12], the authors use the linear-algebraic method to list-decode folded Reed-Solomon codes when the folding parameter has order at least the dimension of the code.

Paper Organization. In Section 2, we collect notation and definitions which will be used throughout the paper. In Section 3, we define and construct "(s, A, t)-subspace designs," which is the new twist on the subspace designs of [9] that drives our results. In Section 4, we show how these subspace designs can be used to construct list-decodable rank-metric codes. In Section 5, we give a list-decodable subcode of folded Reed-Solomon codes with low folding order. The construction of list-decodable subspace codes appears as Appendix A.

We conclude in Section 6 with some open problems.

2 Notation and Definitions

Throughout the presentation of rank-metric codes, $\mathbb{F}_h$ is a finite field of constant size. $\mathbb{F}_q := \mathbb{F}_{h^t}$ extends $\mathbb{F}_h$, and we will think of $\mathbb{F}_q$ as a vector space over $\mathbb{F}_h$ by fixing a basis. We will also have $n = mt$, and the field $\mathbb{F}_{h^n} := \mathbb{F}_{q^m} = \mathbb{F}_{h^{mt}}$ extending $\mathbb{F}_q$.

In our final applications, s will be $\approx 1/\varepsilon$, m will be $\approx s/\varepsilon$, where for rate R, we will be list decoding up to error fraction $(1 - R - \varepsilon)$, and t will grow.

We will be talking about subspaces over a field and its extension, so to avoid any confusion about the underlying field, we will usually refer to a subspace over a field $\mathbb{F}$ as an $\mathbb{F}$-subspace.

We recall some of the definitions of the pseudorandom objects concerning subspaces that we require.

▶ **Definition 1** (Strong subspace designs, [14])**.** A collection S of $\mathbb{F}_q$-subspaces $H_1, \ldots, H_M \subseteq \mathbb{F}_q^m$ is called a (s, A) *subspace design* if for every $\mathbb{F}_q$-linear space $W \subset \mathbb{F}_q^m$ of dimension s,

$$\sum_{i=1}^{M} \dim_{\mathbb{F}_q}(H_i \cap W) \leqslant A.$$

▶ **Definition 2** (Subspace-evasive sets, [12])**.** A subset $\mathcal{V} \subseteq \mathbb{F}_q^k$ is (s, L) *subspace-evasive* if for every $\mathbb{F}_q$-subspace $S \subset \mathbb{F}_q^k$ of dimension s, $|S \cap \mathcal{V}| \leqslant L$.

3 Subspace Designs

Throughout this section q and h will be prime powers with $q = h^t$. In what follows, we will think of subspaces $W \subseteq \mathbb{F}_q^m$ as $\mathbb{F}_h$-subspaces of $\mathbb{F}_h^{mt}$ via some fixed basis embedding.

▶ **Definition 3.** A collection S of $\mathbb{F}_h$-subspaces $H_1, \ldots, H_M \subseteq \mathbb{F}_h^{tm}$ is called a (s, A, t) $\mathbb{F}_h$-*subspace design* if for every $\mathbb{F}_{h^t}$-linear space $W \subset \mathbb{F}_{h^t}^m$ of dimension s,

$$\sum_{i=1}^{M} \dim_{\mathbb{F}_h}(H_i \cap W) \leqslant A.$$

Note that in the above definition the dimension of the input W is measured as a subspace over $\mathbb{F}_{h^t}$ whereas for the intersection, which is an $\mathbb{F}_h$-subspace, the dimension is over $\mathbb{F}_h$.

▶ Remark. When $t = 1$, these are the (strong) subspace designs of [9]. We will be interested in settings where $t = \omega(1)$, so that considering W as a subspace of dimension st over $\mathbb{F}_h$ will generally not give strong enough bounds.

3.1 Existential Bounds

The following proposition shows that good subspace designs exist; indeed, a random collection of subspaces works with high probability. The case $t = 1$ was established in [9].

▶ **Proposition 4.** *Let $\varepsilon > 0$. Let S consist of $M = h^{\varepsilon tm/8}$ $\mathbb{F}_h$-subspaces of codimension εtm in $\mathbb{F}_h^{mt}$, chosen independently at random. Then for any $s < m\varepsilon/2$, with probability at least $1 - q^{-ms}$, S is a $(s, 8s/\varepsilon, t)$ $\mathbb{F}_h$-subspace design. (Here $q = h^t$.)*

Proof. Set $\ell = 8s/\varepsilon$, and let $S = \{H_1, \ldots, H_M\}$. For a fixed $\mathbb{F}_{h^t}$ subspace W of dimension s and any j, the probability that $\dim_{\mathbb{F}_h}(W \cap H_j) \geqslant a$ at most $q^{sa} \cdot q^{-\varepsilon ma} \leqslant q^{-\varepsilon ma/2}$, by assumption on s.

Since the H_i are independent, for a fixed tuple $(a_1, \ldots, a_M)$ of nonnegative integers summing to $\ell = 8s/\varepsilon$, the probability that $\dim(W \cap H_j) \geqslant a_j$ for each j is at most $q^{-\varepsilon m\ell/2} = q^{-4ms}$. Union bounding over the at most q^{ms} choices of W and $\binom{\ell+M}{\ell} \leqslant M^{2\ell}$ choices of $(a_1, \ldots, a_M)$, the probability S is *not* a $(s, 8s/\varepsilon, t)$ $\mathbb{F}_h$-subspace design is at most

$$q^{ms} M^{2\ell} \cdot q^{-4ms} = q^{ms} \cdot q^{2ms} \cdot q^{-4ms} \leqslant q^{-ms} \ . \qquad \blacktriangleleft$$

3.2 Constructive Bounds

In this section, we show how to construct an explicit large $\big(s, 2(m-1)s/\varepsilon, t\big)$ $\mathbb{F}_h$-subspace design consisting of $\mathbb{F}_h$-subspaces of $\mathbb{F}_h^{tm}$ of codimension $2\varepsilon tm$.

The idea, which is natural in hindsight, is to first use a subspace design over $\mathbb{F}_{h^t}$ to ensure that the intersection with any $\mathbb{F}_{h^t}$-subspace of dimension s has low dimension over $\mathbb{F}_{h^t}$, and then to use a subspace-evasive set to reduce the dimension further over $\mathbb{F}_h$. The final construction appears as Theorem 8.

3.2.1 Explicit Subspace-evasive Sets

We first describe the construction of explicit subspace-evasive sets which we will be using.

Let $q > h^{m-1}$, and let $\gamma_1, \ldots, \gamma_m$ be distinct elements of $(\mathbb{F}_q)^*$. Let A be the $s \times m$ matrix with $A_{ij} = \gamma_j^i$. Then Dvir and Lovett [1] showed the following:

▶ **Theorem 5.** *Let $1 \leqslant s \leqslant m$. Let $d_1 > d_2 > \cdots > d_m \geqslant 1$ be integers. Define $f_1, \ldots, f_s \in \mathbb{F}_q[X_1, \ldots, X_m]$ as follows:*

$$f_i(x_1, \ldots, x_m) = \sum_{j=1}^{m} A_{ij} x_j^{d_j} \ . \tag{1}$$

Then:

- *The variety $\mathbf{V} = \{x \in \overline{F}_q^m \mid f_1(x) = \cdots = f_s(x) = 0\}$ satisfies $|\mathbf{V} \cap H| \leqslant (d_1)^s$ for all s-dimensional affine subspaces $H \subset \overline{\mathbb{F}}_q^m$.*
- *If at least s of the degrees d_i are relatively prime to $q-1$, then $|\mathbf{V} \cap \mathbb{F}_q^m| = q^{m-s}$.*

Additionally, the product set $(\mathbf{V} \cap \mathbb{F}_q^m)^{n/m} \subseteq \mathbb{F}^n$ is $(k, (d_1)^k)$-subspace evasive for all $k \leqslant s$.

The below statement follows immediately from Theorem 5 and the fact that when the d_j's are powers of h, the polynomials f_i defined in (1) are $\mathbb{F}_h$-linear functions on $\mathbb{F}_q^m$.

▶ **Corollary 6.** *Setting $d_1 = h^{m-1}, d_2 = h^{m-2}, \ldots, d_m = 1$, we obtain an explicit $\mathbb{F}_h$-linear set S of size $q^{(m-s)n/m}$ over $\mathbb{F}_q^n$ which is $(k, h^{(m-1)k})$ subspace-evasive for all $1 \leqslant k \leqslant s$.*

▶ Remark. One can improve on the degree bounds and therefore the final intersection size via a standard subspace-evasive set without the $\mathbb{F}_h$-linearity requirement. For example, [1] gives a construction of a (non-linear) $\left(s, (s/\varepsilon)^s\right)$ subspace-evasive set over $\mathbb{F}^n$ of size $|\mathbb{F}|^{(1-\varepsilon)n}$.

However, especially in applications for rank-metric codes, linearity is a property which is desirable and often necessary.

3.2.2 Combining with Subspace Designs

The following theorem shows how to achieve our initial goal of ensuring small intersection dimension over the larger field $\mathbb{F}_{h^t}$.

▶ **Theorem 7** ([9])**.** *For $\varepsilon \in (0,1)$, positive integers s, m with $s \leqslant \varepsilon m/4$, and $q > m$, there is an explicit collection of $M = q^{\Omega(\varepsilon m/s)}$ subspaces in $\mathbb{F}_q^m$, each of codimension at most εm, which form a $(s, 2s/\varepsilon, 1)$ $\mathbb{F}_q$-subspace design.*

Combined with Corollary 6, we now have a construction of a $(s, 2(m-1)s/\varepsilon, t)$ $\mathbb{F}_h$-subspace design, summarized in the following statement.

▶ **Theorem 8.** *For integers $s \leqslant \varepsilon m/4$ and $q > m$, there exists an explicit set of $q^{\Omega(\varepsilon m/s)}$ $\mathbb{F}_h$-subspaces in $\mathbb{F}_h^{tm}$ of codimension at most $2\varepsilon tm$ forming a $(s, 2(m-1)s/\varepsilon, t)$ $\mathbb{F}_h$-subspace design.*

Proof. Let $V_1, \ldots, V_M \subseteq \mathbb{F}_q^m$ be the elements of the $(s, 2s/\varepsilon, 1)$ $\mathbb{F}_q$-subspace design of Theorem 7. For each i, define $H_i = V_i \cap S$, where $S \subseteq \mathbb{F}_q^m$ is the $(s, h^{(m-1)s})$ subspace-evasive set of Corollary 6. As S and the V_i's are $\mathbb{F}_h$-linear subspaces, H_i is as well. We claim that the H_i's form the desired $\mathbb{F}_h$-subspace design.

For each i, V_i has codimension εtm, and S has codimension $ts \leqslant \varepsilon tm/4$, so the codimension of H_i is at most $2\varepsilon tm$.

Now let W be an $\mathbb{F}_q$-subspace of dimension s. By the $\mathbb{F}_q$-subspace design property of the V_i's we have

$$\sum_{i=1}^{M} \dim_{\mathbb{F}_q}(V_i \cap W) \leqslant 2s/\varepsilon \ . \tag{2}$$

For each i, we also have that $\dim_{\mathbb{F}_q}(W \cap V_i) = s_i \leqslant s$, so by the subsace evasive property of S from Corollary 6, $W \cap H_i = (W \cap V_i) \cap S$ has at most $h^{(m-1)s_i}$ elements. As $W \cap H_i$ is $\mathbb{F}_h$-linear, we have

$$\dim_{\mathbb{F}_h}(W \cap H_i) \leqslant (m-1)\dim_{\mathbb{F}_q}(W \cap V_i) \ . \tag{3}$$

Combining (2) and (3) we have

$$\sum_i \dim_{\mathbb{F}_h}(W \cap H_i) \leqslant \sum_i (m-1)\dim_{\mathbb{F}_q}(W \cap V_i) \leqslant (m-1)\cdot 2s/\varepsilon \ . \qquad \blacktriangleleft$$

The motivation for constructing the above subspace design is that they yield a subspace that has small intersection with so-called periodic subspaces arising in certain linear-algebraic list decoding algorithms. We recall the definition from [14]. Below, for a string $\mathbf{x} = (x_1, x_2, \ldots, x_\ell)$, we denote by $\mathrm{proj}_{[a,b]}(\mathbf{x})$ the substring $(x_a, x_{a+1}, \ldots, x_b)$.

▶ **Definition 9** (Periodic subspaces)**.** For positive integers s, m, k and $\kappa := mk$, an affine subspace $H \subset \mathbb{F}_q^\kappa$ is said to be (s,m,k)-**periodic** if there exists a subspace $W \subseteq \mathbb{F}_q^m$ of dimension at most s such that for every $j = 1, 2, \ldots, k$, and every prefix $\mathbf{a} \in \mathbb{F}_q^{(j-1)m}$, the projected affine subspace of $\mathbb{F}_q^m$ defined by

$$\{\mathrm{proj}_{[(j-1)m+1,jm]}(\mathbf{x}) \mid \mathbf{x} \in H \text{ and } \mathrm{proj}_{[1,(j-1)m]}(\mathbf{x}) = \mathbf{a}\}$$

is contained in an affine subspace of $\mathbb{F}_q^m$ given by $W + \mathbf{v_a}$ for some vector $\mathbf{v_a} \in \mathbb{F}^m$ dependent on $\mathbf{a}$.

▶ **Proposition 10.** *Let H be a (s,m,k)-periodic affine suspace of $\mathbb{F}_q^{mk}$, and $H_1, H_2, \ldots, H_k \subseteq \mathbb{F}_h^{mt}$ be distinct subspaces from a (s, A, t) $\mathbb{F}_h$-subspace design. Then $H \cap (H_1 \times \cdots \times H_k)$ is an affine subspace over $\mathbb{F}_h$ of dimension at most A.*

Proof. It is clear that $H \cap (H_1 \times \cdots \times H_k)$ is an affine subspace over $\mathbb{F}_h$. Let W be the subspace associated to H as in Definition 9. We will show by induction that $|\mathrm{proj}_{[1,im]}(H) \cap (H_1 \times \cdots \times H_i)| \leqslant h^{\sum_{j=1}^{i} \dim_{\mathbb{F}_h}(W \cap H_j)}$.

In the base case, since H_1 is a subspace, $\mathrm{proj}_{[1,m]}(H) \cap H_1 = (W + v_{\mathbf{0}}) \cap H_1$ is an affine subspace whose underlying subspace lies in $W \cap H_1$. In particular, its size is at most $h^{\dim(W \cap H_1)}$.

Continuing, fix an element $\mathbf{a} \in \mathrm{proj}_{[1,im]}(H) \cap (H_1 \times \cdots \times H_i)$. Because H is periodic and H_{i+1} is linear, the possible extensions of $\mathbf{a}$ in $\mathrm{proj}_{[im+1,(i+1)m]}(H) \cap H_{i+1}$ are given by a coset of $W \cap H_{i+1}$. Thus, there are at most $h^{\dim(W \cap H_{i+1})}$ such extensions. Since by induction there were $h^{\sum_{j=1}^{i} \dim_{\mathbb{F}_h}(W \cap H_j)}$ possibilities for the prefix $\mathbf{a}$, the result follows.

In particular, $H \cap (H_1 \times \cdots \times H_k)$ has dimension over $\mathbb{F}_h$ which is at most $\sum_{i=1}^{k} \dim(W \cap H_i) \leqslant A$, by the subspace design property. $\blacktriangleleft$

4 Explicit List-decodable Rank-metric Codes

In this section, we show how to use the subspace designs of Theorem 8 in order to get explicit list-decodable rank-metric codes of optimal rate for any desired error correction radius.

We first review rank-metric codes, and in particular the Gabidulin code [6], which is the starting point of our construction.

Let h be a prime power, and let $\mathbb{M}_{n\times t}(\mathbb{F}_h)$ be the set of $n \times t$ matrices over $\mathbb{F}_h$. The *rank distance* between $A, B \in \mathbb{M}_{n\times t}(\mathbb{F}_h)$ is $d(A, B) = \operatorname{rank}(A - B)$. A rank-metric code $\mathcal{C}$ is a subset of $\mathbb{M}_{n\times t}(\mathbb{F}_h)$, with rate and distance given by

$$R(\mathcal{C}) = \frac{\log_h|\mathcal{C}|}{nt} \quad \text{and} \quad d(\mathcal{C}) = \min_{A\neq B\in\mathcal{C}}\{d(A, B)\}.$$

The *Gabidulin code* encodes h-linearized polynomials by their evaluations at linearly independent points. Recall that an h-linearized polynomial f over $\mathbb{F}_{h^t}$ is a polynomial of the form $\sum_{i=0}^{\ell} a_i X^{h^i}$, with $a_i \in \mathbb{F}_{h^t}$. If $a_\ell \neq 0$, then ℓ is called the *h-degree* of f. We write $\mathcal{L}_h(t)$ for the set of h-linearized polynomials over $\mathbb{F}_{h^t}$.

Let $0 < k \leqslant n \leqslant t$ be integers, and choose $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_{h^t}$ to be linearly independent over $\mathbb{F}_h$. For every h-linearized polynomial $f \in \mathbb{F}_{h^t}[X]$ of h-degree at most $k - 1$, we can encode f by the column vector $M_f = \big(f(\alpha_1), \ldots, f(\alpha_n)\big)^T$ over $\mathbb{F}_{h^t}$. By fixing a basis of $\mathbb{F}_{h^t}$ over $\mathbb{F}_h$, we can also think of M_f as an $n \times t$ matrix over $\mathbb{F}_h$. This yields the Gabidulin code

$$\mathcal{C}_G(h; n, t, k) := \{M_f \in \mathbb{M}_{n\times t}(\mathbb{F}_h) \mid f \in \mathcal{L}_h(t),\ h\text{-degree}(f) \leqslant k - 1\}.$$

If a rank-metric codeword X is transmitted, and a matrix Y is received, we say that $\operatorname{rank}(Y - X)$ *rank errors* have occurred.

Suppose that $t = nm$ for some integer m, so that $\mathbb{F}_{h^t}$ has a subfield $\mathbb{F}_{h^n} =: \mathbb{F}_q$. In the case when the evaluation points $\alpha_1, \ldots, \alpha_n$ of the Gabidulin code span $\mathbb{F}_{h^n}$, Guruswami and Xing [14] show the following:

▶ **Theorem 11** ([14]). *Let $f \in \mathbb{F}_{h^t}[X]$ be an h-linearized polynomial with h-degree at most $k-1$. Suppose that a codeword $M_f = \big(f(\alpha_1), \ldots, f(\alpha_n)\big)^T$ is transmitted and $Y = (y_1, \ldots, y_n)^T$ is received with at most e rank errors. If $e \leqslant s(n - k)/(s + 1)$, then there is an algorithm running in time $\operatorname{poly}(n, m, \log q)$ outputting a $(s - 1, m, k)$-periodic subspace containing all candidate messages f.*

By Proposition 10, by restricting the message polynomials $f = \sum_i f_i X^{q^i}$ to have coefficients $f_i \in H_{i+1}$ for $0 \leqslant i < k$, where $H_1, H_2, \ldots, H_k$ are distinct elements of the subspace design in Theorem 8, the final list of candidate messages will have dimension at most $2(m - 1)s/\varepsilon$ over $\mathbb{F}_h$, or size at most $h^{2(m-1)s/\varepsilon}$. As one can take $m = O(s/\varepsilon)$ for the necessary subspace design guaranteed by Theorem 8, we can conclude the following theorem, which is our main result.

▶ **Theorem 12.** *For every $\varepsilon > 0$ and integer $s > 0$, there exists an explicit $\mathbb{F}_h$-linear subcode of the Gabidulin code $\mathcal{C}_G(h; n, t, k)$ with evaluation points spanning $\mathbb{F}_{h^n}$ of rate $(1 - 2\varepsilon)k/n$ which is list-decodable from $\frac{s}{s+1} \cdot (n - k)$ rank errors. The final list is contained in an $\mathbb{F}_h$-subspace of dimension at most $O(s^2/\varepsilon^2)$.*

5 Application to Low-order Folding of Reed-Solomon Codes

In this section, we show how the idea of only evading subspaces over an extension field can be used to give an algorithm for list-decoding (subcodes of) folded Reed-Solomon codes in the case when the folding parameter has low ($O(1)$) order.

As in the case of KK codes, our decoding algorithm follows the framework of interpolating a linear polynomial and then solving a linear system for candidate polynomials. Fix γ generating $\mathbb{F}_q^*$. Let $N = \frac{q-1}{\ell}$, and let $\zeta = \gamma^N$, which has order ℓ in $\mathbb{F}_q$. Then the **low-order**

folded Reed-Solomon code encodes a polynomial f of degree $< k$ by

$$f \mapsto \begin{bmatrix} f(1) & f(\gamma) & \cdots & f(\gamma^{N-1}) \\ f(\zeta) & f(\zeta\gamma) & \cdots & f(\zeta\gamma^{N-1}) \\ \vdots & \vdots & \ddots & \vdots \\ f(\zeta^{\ell-1}) & f(\zeta^{\ell-1}\gamma) & \cdots & f(\zeta^{\ell-1}\gamma^{N-1}) \end{bmatrix}.$$

Similarly to folded Reed-Solomon codes, this is a code of rate $\frac{k}{\ell N}$ and distance $N - (k-1)/\ell$.

5.1 Interpolation

Given a received word

$$\begin{pmatrix} y_{00} & y_{01} & \cdots & y_{0(N-1)} \\ y_{10} & y_{11} & \cdots & y_{1(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ y_{(\ell-1)0} & y_{(\ell-1)1} & \cdots & y_{(\ell-1)(N-1)} \end{pmatrix},$$

we would like to interpolate a (nonzero) polynomial

$$Q(X, Y_1, \ldots, Y_s) = A_0(X) + A_1(X)Y_1 + \cdots + A_s(X)Y_s$$

such that

$$Q\big(\gamma^{iN+j}, y_{ij}, y_{(i+1)j}, \ldots, y_{(i+s-1)j}\big) = 0 \qquad i \in \{0, \ldots, \ell-1\},\ j \in \{0, \ldots, N-1\}, \tag{4}$$

where all indices are taken modulo ℓ.

We will require $\deg(A_0) \leqslant D + k - 1$, and $\deg(A_i) \leqslant D$ for $i > 0$.

▶ **Lemma 13.** *Let*

$$D = \left\lfloor \frac{\ell N - k + 1}{s+1} \right\rfloor.$$

Then a nonzero polynomial Q satisfying (4) *exists (and can be found by solving a linear system).*

Proof. The number of interpolation conditions is ℓN. The quantity $(D+1)(s+1) + k - 1 > \ell N$ is the number of degrees of freedom for the interpolation, and the conditions are homogeneous, so a nonzero solution exists. ◀

▶ **Lemma 14.** *If the number of agreements t is greater than $\frac{D+k-1}{\ell}$, then*

$$Q\big(X, f(X), f(\zeta X), \ldots, f(\zeta^{s-1}X)\big) = 0. \tag{5}$$

Proof. $Q\big(X, f(x), \ldots, f(\zeta^{s-1}X)\big)$ is a univariate polynomial of degree $D + k - 1$, and each correct column j yields ℓ distinct roots γ^{iN+j} for $i \in \{0, \ldots, \ell-1\}$. Thus if $t\ell > \deg D + k - 1 \geqslant \deg Q$, Q is the zero polynomial. ◀

For our choice of D, the requirement on t in Lemma 14 is met if t satisfies

$$\frac{t}{N} \geqslant \frac{1}{s+1} + \frac{s}{s+1}R. \tag{6}$$

▶ Remark. In ordinary folded Reed-Solomon codes, where the folding parameter is primitive of order $q-1$, the agreement fraction required to satisfy (5) is

$$\frac{t}{N} \geqslant \frac{1}{s+1} + \frac{s}{s+1}\frac{\ell R}{\ell - s + 1},$$

which is higher than (6). In our case, because ζ has low order, we are able to use interpolation conditions that "wrap around," allowing us to impose ℓ conditions per coordinate rather than $\ell - s + 1$. Therefore we can satisfy Equation (5) with lower agreement. On the other hand, it is known how to list-decode folded Reed-Solomon codes themselves, whereas we are only able to list-decode a subcode.

5.2 Decoding

In this section, we describe how to solve the system

$$Q\big(X, f(X), f(\zeta X), \ldots, f(\zeta^{s-1}X)\big) = 0 \tag{5}$$

for candidate polynomials f.

▶ **Proposition 15.** *Given an irreducible polynomial $R(X) \in \mathbb{F}_q[X]$ such that*

- $\deg R \geqslant k$, *and*
- *for some a, $\zeta X \equiv X^{q^a} \pmod R$.*

Then the set of f of degree $< k$ satisfying (5) *is an $\mathbb{F}_{q^a}$-affine subspace of dimension at most $s-1$.*

Proof. The condition (5) says

$$0 = A_0(X) + A_1(X)f(X) + A_2(X)f(\zeta X) + \cdots + A_s(X)f(\zeta^{s-1}X).$$

Then we have

$$A_0(X) + A_1(X)f(X) + A_2(X)f(X)^{q^a} + \cdots + A_s(X)f(X)^{q^{(s-1)a}} \equiv 0 \pmod R.$$

By dividing out the highest power of R which divides every A_i, Equation (5) is still satisfied and we may assume that this equation is nonzero mod R.

In particular, this equation has at most $q^{(s-1)a}$ solutions for f mod R. When $\deg f < k \leqslant \deg R$, f is uniquely determined by its residue mod R and there are at most $q^{(s-1)a}$ solutions for f.

The fact that the solution space is $\mathbb{F}_{q^a}$-affine follows from the fact that the terms in which $f(X)$ appears all have degree q^{ai} for some i. ◀

Because the output space is a subspace (over the large field $\mathbb{F}_{q^a}$), by picking the message polynomials f to come from a subspace-evasive set, we can reduce the list size bound. More specifically, if ℓ is at least s/ε, [1] gives a construction of a $(s, (s/\varepsilon)^s)$ subspace-evasive set S over $(\mathbb{F}_{q^a})^{k/a}$ of size $q^{(1-\varepsilon)k}$. By precoding the messages to come from this set S, we are able to both encode and compute the intersection of the code with the output subspace of Proposition 15 in polynomial time.

Setting $s = O(1/\varepsilon)$ and $\ell = O(s/\varepsilon)$, we obtain the following.

▶ **Corollary 16.** *For every $\varepsilon > 0$ and $R \in (0,1)$, there is an explicit rate R subcode of a low-order folded Reed-Solomon code which is list-decodable from a $1 - R - \varepsilon$ fraction of errors with list size $(1/\varepsilon)^{O(1/\varepsilon)}$, given an irreducible polynomial satisfying the conditions of Proposition 15.*

▶ Remark. By using Corollary 6 instead of the results of [1], we can give a similar guarantee which yields a *linear* subcode, but with a larger list size guarantee of $q^{\mathrm{poly}(1/\varepsilon)}$.

The techniques of [14] using subspace designs could also be applied directly to the case of low-order folding, with a resulting list size of $n^{\mathrm{poly}(1/\varepsilon)}$. We are able to get an improvement using the observation that the space of candidates is actually a low-dimensional subspace over a much larger field.

5.3 Constructing High-degree Irreducibles

The decoding algorithm of the previous section relied on working modulo a high-degree irreducible factor of $X^{q^a} - \zeta X$. In what follows, we consider the problem of finding such a factor efficiently.

▶ **Proposition 17.** *For $\zeta \in \mathbb{F}_q$ of order ℓ, the irreducible factors over $\mathbb{F}_q[X]$ of*

$$X^{q^a-1} - \zeta$$

have degree dividing $a\ell$. In particular, all roots of $X^{q^a-1} - \zeta$ lie in $\mathbb{F}_{q^{a\ell}}$.

Proof. As $X^{(q^a-1)\ell} \equiv 1 \pmod{X^{q^a-1} - \zeta}$, it is enough to see that $(q^a - 1)\ell$ divides $q^{a\ell} - 1$. This implies that $X^{q^a-1} - \zeta$, and thus all of its irreducible factors, divides $X^{q^{a\ell}} - X$.

As $\ell \mid q - 1$, we have

$$\frac{q^{a\ell} - 1}{q^a - 1} = q^{a(\ell-1)} + q^{a(\ell-2)} + \cdots + q^a + 1 \equiv 0 \pmod{\ell} . \qquad \blacktriangleleft$$

▶ **Corollary 18.** *If a and ℓ with $a > 2\ell$ are distinct primes, at least half of the roots of $X^{q^a-1} - \zeta$ have irreducible polynomials of degree $a\ell$.*

Proof. By Proposition 17, all of the irreducible factors of $X^{q^a-1} - \zeta$ have degrees in the set $\{1, a, \ell, a\ell\}$. No irreducible factor has degree 1 or a, because any irreducible of degree 1 or a divides $X^{q^a-1} - 1$ and therefore does not divide $X^{q^a-1} - \zeta$ for $\zeta \neq 1$.

Because $X^{q^a-1} - \zeta$ has no repeated factors, it has at most q^ℓ roots which lie in $\mathbb{F}_{q^\ell}$ (and hence have irreducible polynomials of degree ℓ.

Thus, under the assumptions on a and ℓ, $X^{q^a-1} - \zeta$ has at least $(q^a - q^\ell - 1) \geqslant q^\ell$ roots of degree $a\ell$. Thus at least half of of $X^{q^a-1} - \zeta$'s roots have irreducible polynomials of degree $a\ell$. $\blacktriangleleft$

In particular, by choosing a to be a prime in the range $[k/\ell, 2k/\ell]$, we have $k \leqslant a\ell \leqslant 2k$, so that an irreducible factor of $X^{q^a-1} - \zeta$ will satisfy the conditions of Proposition 15. The next section will show that we cannot hope to improve much on the value of a.

Given a value for a for which $X^{q^a-1} - \zeta$ has many degree $a\ell$ factors, the problem remains to compute one. In what follows, we describe one randomized approach.

Recall that a and ℓ are primes, and that we are trying to find a degree $a\ell$ factor of $X^{q^a-1} - \zeta$. The idea is to sample a root of $X^{(q^a-1)\ell} - 1$. Consider the following procedure:

1. Sample $\beta \in (\mathbb{F}_{q^a})^*$ uniformly at random.
2. Compute the roots $\rho_1, \ldots, \rho_\ell$ of $X^\ell - \beta$, which lie in $\mathbb{F}_{q^{a\ell}}$ by Proposition 17. This can be done in time $\tilde{O}(n^2 \log(q^a) \log^{-1} \varepsilon)$ with failure probability ε using a variant of Berlekamp's algorithm (see, for example, [15]).
3. Compute $\rho_i^{q^a-1}$ for each i and output the minimal polynomial of ρ_i over $\mathbb{F}_q$ if $\rho_i^{q^a-1} = \zeta$.

First note that steps 1–2 sample each root of $X^{(q^a-1)\ell}-1$ uniformly. Each ρ_i computed in step 2 satisfies $\rho_i^\ell \in (\mathbb{F}_{q^a})^*$, so ρ_i is a root of $X^{(q^a-1)\ell}-1$. Conversely, each nonzero β yields ℓ distinct roots of $X^\ell - \beta$, which are distinct for distinct β, yielding $(q^a-1)\ell$ roots.

Therefore, with probability $1/\ell$, we will find a root ρ of $X^{q^a-1}-\zeta$. By Corollary 18, ρ's minimal polynomial has degree $a\ell$ with probability at least $1/2$.

We can thus conclude that, with probability at least $\frac{1}{2\ell}-\varepsilon$, we find an irreducible factor of $X^{q^a-1}-\zeta$ of degree $a\ell$.

5.4 Relationship to Reed-Solomon List-decoding

The original motivation for studying low-order folding was the following reduction from Reed-Solomon codes.

Given a polynomial f of degree $< k/\ell$ evaluated at distinct points $1, \gamma^\ell, \gamma^{2\ell}, \ldots, \gamma^{N\ell}$, we can think of it as a degree $< k$ polynomial $g(X) = f(X^\ell)$. For ζ of order ℓ, we have that $g(\zeta^i X) = g(X)$ for every i. In particular, the associated low-order folded Reed-Solomon codeword encoding $g(X)$ is simply

$$\begin{bmatrix} f(1) & f(\gamma^\ell) & \ldots & f(\gamma^{N\ell}) \\ f(1) & f(\gamma^\ell) & \ldots & f(\gamma^{N\ell}) \\ \vdots & \vdots & \ddots & \vdots \\ f(1) & f(\gamma^\ell) & \ldots & f(\gamma^{N\ell}) \end{bmatrix}. \tag{7}$$

Notice that if $f(\gamma^{i\ell})$ is correct, then the entire ith column is correct, so an algorithm to list-decode the low-order folded RS code from an η fraction of errors will also list-decode the Reed-Solomon code with evaluation points $(1, \gamma^\ell, \ldots, \gamma^{N\ell})$ from the same error fraction.

This reduction also helps to show that the precoding used to conclude Corollary 16 is necessary for a polynomial list size. To see this, consider the behavior of the algorithm on a transmitted codeword as in Equation (7). If there is enough agreement, the algorithm will interpolate polynomials $A_i(X)$ satisfying

$$0 = A_0 + A_1(X)g(X) + A_2(X)g(\zeta X) + \cdots + A_s(X)g(\zeta^{s-1}X) \tag{8}$$

$$= A_0(X) + g(X)\sum_{i=1}^{s} A_i(X). \tag{9}$$

If $\sum_{i>0} A_i(X) \neq 0$, then $g(X)$, and thus $f(X)$, can be recovered *uniquely* by computing $A_0(X)/\sum_{i>0} A_i(X)$; however, this will not be possible in general outside of the unique decoding radius. If $\sum_{i>0} A_i(X)$ is 0, then $A_0(X) = 0$ as well and *any* function which is a polynomial of X^ℓ satisfies Equation (9), and in particular the output list must have size at least $q^{k/\ell}$. Recall that ℓ is a constant in our application.

This implies that without precoding, the dimension of the list output by Proposition 15 over $\mathbb{F}_q$ must be $\Omega(k/\ell)$. Note that for the value $a = \theta(k/\ell)$ found in Section 5.3, the list size before precoding would be $O(ks/\ell)$.

6 Conclusion and Open Questions

We have given an explicit construction of list-decodable rank-metric and subspace codes, which were obtained by restricting known codes to carefully chosen subcodes. However, our results give no insight into whether the Gabidulin and KK codes can be themselves list-decoded beyond half the minimum distance. We close with the following natural open problems.

- Is it combinatorially feasible to list-decode Gabidulin codes *themselves* beyond half the distance? We note that it was recently shown that there is no analog of the classical Hamming-metric Johnson bound in the world of rank-metric codes always guaranteeing list-decodability beyond half the minimum distance [24]. Therefore, a proof of list-decodability past the unique decoding radius (say for the Gabidulin code) must account for the code structure beyond just the minimum distance.
- Assuming it is combinatorially feasible, can we give an efficient algorithm to list-decode Gabidulin codes without using subcodes or special evaluation points?
- Currently, for rate R codes, we do not know where in the range $(1-\sqrt{R}, 1-R)$ the list-decoding radius of Reed-Solomon codes lies, and where in the range $[(1-R)/2, 1-\sqrt{R}]$ the list-decoding radius of Gabidulin codes lies. Is there a relationship between these questions?
- Can one construct better subspace-evasive sets to give an *explicit* code that is list-decodable from a fraction $1-R-\varepsilon$ of errors with $\mathrm{poly}(1/\varepsilon)$ list-size? We only known a list-size upper bound that is exponential in $1/\varepsilon$ for current explicit constructions, whereas a list-size of $O(1/\varepsilon)$ can be obtained with a Monte Carlo construction [12, 13, 14]. This question is open for errors in the usual Hamming metric also.

Acknowledgment. We thank Antonia Wachter-Zeh for bringing to our attention the lack of a Johnson-type bound for list decoding rank-metric codes [24].

References

1 Z. Dvir and S. Lovett. Subspace evasive sets. In *Proceedings of the 44th ACM Symposium on Theory of Computing*, pages 351–358, 2012.

2 T. Etzion and N. Silberstein. Error-correcting codes in projective spaces via rank-metric codes and ferrers diagrams. *IEEE Transactions on Information Theory*, 55:2909–2919, 2009.

3 T. Etzion and A. Vardy. Error-correcting codes in projective space. *IEEE Transactions on Information Theory*, 57:1165–1173, 2011.

4 C. Faure. Average number of Gabidulin codewords within a sphere. In *Int. Workshop on Alg. Combin. Coding Theory (ACCT)*, pages 86–89, 2006.

5 M. A. Forbes and A. Shpilka. On identity testing of tensors, low-rank recovery and compressed sensing. In *Proceedings of the 44th ACM Symposium on Theory of Computing*, pages 163–172, 2012.

6 E. M. Gabidulin. Theory of codes with maximal rank distance. *Problems of Information Transmission*, 21(7):1–12, 1985.

7 E. M. Gabidulin. A fast matrix decoding algorithm for rank-error-correcting codes. In G. D. Cohen, S. Litsyn, A. Lobstein, and G. Zémor, editors, *Algebraic Coding*, volume 573 of *Lecture Notes in Computer Science*, pages 126–133. Springer, 1991.

8 E. M. Gabidulin, A. V. Paramonov, and O. V. Tretjakov. Ideals over a non-commutative ring and their applications in cryptology. In D. W. Davies, editor, *EUROCRYPT*, volume 547 of *Lecture Notes in Computer Science*, pages 482–489. Springer, 1991.

9 V. Guruswami and S. Kopparty. Explicit subspace designs. In *Proceedings of the 54th IEEE Symposium on Foundations of Computer Science*, 2013.

10 V. Guruswami, S. Narayanan, and C. Wang. List decoding subspace codes from insertions and deletions. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 183–189, January 2012.

11 V. Guruswami and A. Rudra. Explicit codes achieving list decoding capacity: Error-correction with optimal redundancy. *IEEE Transactions on Information Theory*, 54(1):135–150, 2008.

12 V. Guruswami and C. Wang. Linear-algebraic list decoding for variants of Reed-Solomon codes. *IEEE Transactions on Information Theory*, 59(6):3257–3268, 2013.

13 V. Guruswami and C. Xing. Folded codes from function field towers and improved optimal rate list decoding. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:36, 2012. Extended abstract appeared in the *Proceedings of the 44th ACM Symposium on Theory of Computing (STOC'12)*.

14 V. Guruswami and C. Xing. List decoding Reed-Solomon, Algebraic-Geometric, and Gabidulin subcodes up to the Singleton bound. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:146, 2012. Extended abstract appeared in the *Proceedings of the 45th ACM Symposium on Theory of Computing (STOC'13)*.

15 E. Kaltofen. Polynomial factorization 1987–1991. *Proceedings of LATIN '92, LNCS*, 583:294–313, 1992.

16 R. Koetter and F. R. Kschischang. Coding for errors and erasures in random network coding. *IEEE Transactions on Information Theory*, 54(8):3579–3591, 2008.

17 P. Loidreau. Designing a rank metric based McEliece cryptosystem. In N. Sendrier, editor, *PQCrypto*, volume 6061 of *Lecture Notes in Computer Science*, pages 142–152. Springer, 2010.

18 H. Lu and P. V. Kumar. A unified construction of space-time codes with optimal rate-diversity tradeoff. *IEEE Transactions on Information Theory*, 51(5):1709–1730, 2005.

19 P. Lusina, E. M. Gabidulin, and M. Bossert. Maximum rank distance codes as space-time codes. *IEEE Transactions on Information Theory*, 49(10):2757–2760, 2003.

20 H. Mahdavifar and A. Vardy. Algebraic list-decoding on the operator channel. In *Proceedings of the IEEE International Symposium on Information Theory*, pages 1193–1197, 2010.

21 H. Mahdavifar and A. Vardy. List-decoding of subspace codes and rank-metric codes up to Singleton bound. *CoRR*, abs/1202.0866, 2012.

22 R. M. Roth. Maximum-rank array codes and their application to crisscross error correction. *IEEE Transactions on Information Theory*, 37(2):328–336, 1991.

23 D. Silva, F. R. Kschischang, and R. Koetter. A rank-metric approach to error control in random network coding. *IEEE Transactions on Information Theory*, 54(9):3951–3967, 2008.

24 A. Wachter-Zeh. Bounds on list decoding of rank-metric codes. *IEEE Transactions on Information Theory*, 59(11):7268–7277, 2013.

A Explicit List-decodable Subspace Codes

A.1 The Operator Channel and Subspace Codes

For a vector space W, let $\mathcal{P}(W)$ denote the set of all subspaces of W, and $\mathcal{P}_n(W)$ the set of all n-dimensional subspaces of W.

We recall the definition of the operator channel from [16].

▶ **Definition 19.** An *operator channel* C associated with the *ambient space* W is a channel with input and output alphabet $\mathcal{P}(W)$. The channel input V and output U are related by

$$U = \mathcal{H}_k(V) + E,$$

where $k = \dim(U \cap V)$, E is an error subspace (wlog E may be taken such that $E \cap V = \{0\}$), and $\mathcal{H}_k(V)$ is an operator returning an arbitrary k-dimensional subspace of V.

In transforming V to U, we say that operator channel commits $r = \dim(V) - k$ *deletions* and $t = \dim(E)$ *insertions*.

A *subspace code* C is a subset of $\mathcal{P}_n(\mathbb{F}_q^t)$ for some n. We define the *rate* of a subspace code to be

$$R(C) = \frac{\log_q |C|}{nt}.$$

A.2 The Kötter-Kschischang (KK) Code

Our constructions will be subcodes of the KK code (as introduced in [16]), which we now define.

For n dividing t, let $\mathbb{F}_{h^t}$ extend $\mathbb{F}_h$, and let $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_{h^t}$ generate the subfield $\mathbb{F}_{h^n} := \mathbb{F}_q$.

Set $m = t/n$. Then the (n, k, t) **KK code** encodes an $\mathbb{F}_h$-linearized polynomial over $\mathbb{F}_{q^m} = \mathbb{F}_{h^t}$ of q-degree $< k$ by

$$f(X) \mapsto \text{span}\{(\alpha_i, f(\alpha_i)\}_{i=1}^n.$$

The encoding of f is an n-dimensional vector space in the ambient space of dimension $n + t$ over $\mathbb{F}_h$.

When $k < n$, this code has distance $2(n - k + 1)$ and rate

$$\frac{\log_h q^{mk}}{n(n+t)} = \frac{k}{n}\left(\frac{1}{1+n/t}\right) \approx \frac{k}{n} \quad (\text{when } n \ll t).$$

If the channel commits $\leqslant \mu$ deletions and $\leqslant \rho$ insertions, where $s\mu + \rho < s(n - k + 1)$, Guruswami and Xing [14] give a list-decoding algorithm which outputs a $(s-1, m, k)$-periodic subspace in $\mathbb{F}_q^{mk}$ containing all candidate messages.

A.3 List-decodable Subcodes

By restricting the coefficients of the message polynomial f to come from distinct $H_1, \ldots, H_k$ from the $\big(s, 2(m-1)s/\varepsilon, t\big)$-subspace design of Theorem 8, and setting $m \approx s/\varepsilon$, we can prune the list down to a $\mathbb{F}_h$-subspace of dimension $O(s^2/\varepsilon^2)$.

Notice that the H_i's are $\mathbb{F}_h$-linear subspaces, so the restricted subcode is linear. In summary, we have:

▶ **Theorem 20.** *For every $\varepsilon > 0$ and integer $s > 0$, there exists an explicit linear subcode of the $\big(n, k, sn/\varepsilon\big)$ KK code of rate $(1 - \varepsilon)k/n$ which is list-decodable from ρ insertions and μ deletions, provided $\rho + s\mu < s(n - k + 1)$.*

Moreover, the output list is contained in an $\mathbb{F}_h$-subspace of dimension $O(s^2/\varepsilon^2)$.

Exchangeability and Realizability: De Finetti Theorems on Graphs

T. S. Jayram and Jan Vondrák

IBM Almaden Research Center, San Jose, CA, USA
{jayram,jvondrak}@us.ibm.com

Abstract

A classic result in probability theory known as de Finetti's theorem states that exchangeable random variables are equivalent to a mixture of distributions where each distribution is determined by an i.i.d. sequence of random variables (an "i.i.d. mix"). Motivated by a recent application in [18] and more generally by the relationship of local vs. global correlation in randomized rounding, we study weaker notions of exchangeability that still imply the conclusion of de Finetti's theorem. We say that a bivariate distribution ρ is G-realizable for a graph G if there exists a joint distribution of random variables on the vertices such that the marginal distribution on each edge equals ρ.

We first characterize completely the G-realizable distributions for all symmetric/arc-transitive graphs G. Our main results are forms of de Finetti's theorem for general graphs, based on spectral properties. Let $\lambda_1(G) \geq \ldots \geq \lambda_n(G)$ denote the eigenvalues of the adjacency matrix of G.

1. We prove that if ρ is G_n-realizable for a sequence of graphs such that $\lim_{n\to\infty} \frac{\lambda_n(G_n)}{\lambda_1(G_n)} = 0$, then ρ is described by a probability matrix that is positive-semidefinite. For random variables on domains of size $|\mathcal{D}| \leq 4$, this implies that ρ must be an i.i.d. mix.
2. If ρ is G_n-realizable for a sequence of (n, d, λ)-graphs G_n (d-regular with all eigenvalues except for one bounded by λ in absolute value) such that $\lim_{n\to\infty} \frac{\lambda(G_n)}{d(G_n)} = 0$, then ρ is an i.i.d. mix.
3. If ρ is $\vec{G}_n$-realizable for a sequence of directed graphs such that each of them is an arbitrary orientation of an (n, d, λ)-graph G_n, and $\lim_{n\to\infty} \frac{\lambda(G_n)}{d(G_n)} = 0$, then ρ is an i.i.d. mix.

1998 ACM Subject Classification G.3 Probability and Statistics

Keywords and phrases exchangeability, de Finetti's Theorem, spectral graph theory, regularity lemma

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.762

1 Introduction

De Finetti's theorem [3] is a classic result in probability theory and statistics which states that exchangeable observations are equivalent to independent observations conditioned on a latent variable. Formally, a finite sequence of random variables $X_1, X_2, \ldots, X_n$ (or their joint distribution) is said to be *exchangeable* if their joint distribution is the same as that of the sequence $X_{\pi(1)}, X_{\pi(2)}, \ldots, X_{\pi(n)}$, for all permutations π on $[n]$. Note that every i.i.d. sequence satisfies this property and so does every convex combination of distributions determined by i.i.d. sequences (over the same domain). Following [21], we call such a distribution an *i.i.d. mix*. An infinite sequence of random variables is said to be exchangeable if every finite prefix is itself exchangeable. De Finetti's theorem states that every infinite exchangeable sequence is equivalent to an i.i.d. mix (of infinite sequences). Originally shown for Bernoulli random variables [4], it is now part of a broad theory of exchangeability; Kallenberg's book [8] is an excellent reference.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 762–778

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

Diaconis [5] was the first one to study exchangeability for finite sequences, and observe that de Finetti's theorem fails, e.g. for length two.[1] Diaconis and Freedman [6] considered a stronger form of exchangeability. Say that an exchangeable sequence $X_1, X_2, \ldots, X_k$ is *n-extendable*, for $n \geq k$, if it can be extended to an exchangeable sequence $X_1, X_2, \ldots, X_n$. Diaconis and Freedman [6] showed that every k-variate n-extendable distribution ρ is at variation distance at most $\frac{k(k-1)}{2n}$ from an i.i.d. mix, independent of the size of the domain of each X_i. For finite domains of size d, they showed a bound of $2dk/n$. This can be improved to $O(k\sqrt{d}/n)$ [19]. Here, we mostly deal with finite domains unless stated otherwise.

In recent work [18], a related question arose in the design of randomized rounding schemes for the Multiway Cut problem. A bivariate distribution ρ was defined in [18] to be *pairwise realizable*, if for each n there exist $X_1, X_2, \ldots, X_n$ such that the marginal distribution of (X_i, X_j) for every distinct $i, j \in [n]$ equals ρ. The authors showed that a distribution ρ is pairwise realizable if and only if ρ is an i.i.d. mix. Although this statement does not appear to have been formulated explicitly before, it is implicit in several works in the area of exchangeability (e.g. [21]). We observe here that it can be actually derived directly from the Diaconis-Freedman theorem (more generally for k-variate distributions and with the same quantitative bounds; see the last part of this section for details).

Since pairwise realizability is sufficient to derive the conclusion of the Diaconis-Freedman theorem (that such a bivariate distribution must be close to an i.i.d. mix), it is natural to ask whether the assumption could be weakened even further. In particular, is it necessary to assume that all pairs have the same distribution to conclude that this distribution is close to an i.i.d. mix? More precisely, we investigate the following concept.

G-realizable Distributions. We say that a bivariate distribution ρ is G-realizable for an undirected graph G if there exists a joint distribution of variables $\{X_v : v \in V(G)\}$ such that the distribution of (X_u, X_v) is ρ for each edge $\{u, v\} \in E(G)$. Similarly, ρ is $\vec{G}$-realizable for a directed graph $\vec{G}$ if the marginal distribution of (X_u, X_v) for each directed edge (u, v) is ρ. Note that the condition of being pairwise realizable is equivalent to being K_n-realizable for every n. The question we ask is, what distributions are G-realizable for a given graph G, and how does this depend on the structure of G? In particular, what kinds of graphs admit non-trivial G-realizable distributions and what kinds of graphs admit only i.i.d. mixes similar to de Finetti's theorem?

Motivation. Apart from pure mathematical curiosity, we are motivated by the question of *local vs. global correlation* in algorithm design [2, 1]. Generally speaking, mathematical relaxations of combinatorial optimization problems assign fractional values to elements or small subsets of the ground set, which can be interpreted as probabilities of certain local configurations. Hence a fractional solution can be viewed as a collection of local distributions. The question is whether these distributions can be realized globally, on the entire ground set. Usually this is possible only with some loss in the objective function, which leads to the design of approximation algorithms. In this paper, we study the basic question of characterizing the local distributions that can be realized globally for all pairs given by a certain graph.

[1] Take the distribution $(X, Y) \sim \rho$ on $\{0, 1\}^2$ given by $\Pr[X = 1, Y = 0] = \Pr[X = 0, Y = 1] = 1/2$. In particular, $\Pr[X = Y] = 0$. However, for any pair (U, V) of i.i.d. Bernoulli variables, $\Pr[U = V] \geq \frac{1}{2}$. Therefore, ρ cannot be a mix of i.i.d. distributions.

Our Results. First, we establish some basic properties of G-realizability. We show that it suffices to consider the *core* of G, that is, a minimal induced subgraph H of G such that H and G are homomorphically equivalent, whence being G-realizable is equivalent to being H-realizable. In particular, G-realizability for a bipartite graph G does not impose any restrictions at all. For arc-transitive directed graphs and for the analogous property of symmetric undirected graphs, we completely characterize the class of distributions that are G-realizable.

Our main results can be viewed as variants of the Diaconis-Freedman theorem (for $k = 2$), under weaker assumptions. For example, if ρ is K_n-realizable, the Diaconis-Freedman theorem shows that the distribution tends to an i.i.d. mix as $n \to \infty$. We show that even realizability on much sparser graphs leads to the same conclusion.

Realizability on Graphs with Spectral Properties. In the following, $\lambda_1 \geq \ldots \geq \lambda_n$ denote the eigenvalues of the adjacency matrix of G. For comparison with the Diaconis-Freedman theorem, recall that the eigenvalues of K_n are $\lambda_1 = n - 1$ and $\lambda_2 = \ldots = \lambda_n = -1$.

1. We prove that if ρ is G_n-realizable for a sequence of graphs such that $\lim_{n\to\infty} \frac{\lambda_n(G_n)}{\lambda_1(G_n)} = 0$, then ρ is described by a probability matrix that is positive-semidefinite. For random variables on domains of size $|\mathcal{D}| \leq 4$, this implies that ρ must be an i.i.d. mix.
2. If ρ is G_n-realizable for a sequence of (n, d, λ)-graphs G_n (d-regular with all eigenvalues except for one bounded by λ in absolute value) such that $\lim_{n\to\infty} \frac{\lambda(G_n)}{d(G_n)} = 0$, then ρ is an i.i.d. mix.
3. If ρ is $\vec{G}_n$-realizable for a sequence of directed graphs such that each of them is an arbitrary orientation of an (n, d, λ)-graph G_n, and $\lim_{n\to\infty} \frac{\lambda(G_n)}{d(G_n)} = 0$, then ρ is an i.i.d. mix.

Let us discuss some aspects of the above results. It is easy to see that being realizable on a bipartite graph G does not impose any restrictions at all and hence some condition forbidding bipartiteness is necessary to derive any non-trivial result. Result (1) applies to graphs that are "far from bipartite" in the sense that the normalized minimum eigenvalue is close to 0. (See [20] for an explicit connection.) More precisely (see Section 3.1), our result states that every G-realizable distribution is within a $|\frac{\lambda_n(G)}{\lambda_1(G)}|$-distance of a distribution whose probability matrix is positive semidefinite. For domains of size up to 4, this imples that ρ must be close to an i.i.d. mix (due to a connection between doubly nonnegative and completely positive matrices which we discuss in Section 3.2). It is an interesting question whether there is a distribution on a domain larger than 4 that is not an i.i.d. mix and realizable on a sequence of graphs such that $\lim_{n\to\infty} \frac{\lambda_n(G_n)}{\lambda_1(G_n)} = 0$.

Result (2) applies to all finite domains, by making a stronger assumption: Here, the distribution should be realizable on a family of *pseudorandom graphs*, defined in terms of normalized eigenvalues (see Section 3.3). The quantitative bound that we prove here is that a distribution on domain $\mathcal{D}$ realizable on an (n, d, λ)-graph must be $\frac{\lambda}{d}|\mathcal{D}|$-close to an i.i.d. mix. In contrast to the Diaconis-Freedman theorem, we have a dependence on $|\mathcal{D}|$ here which seems to be necessary. We show that for any (fixed) symmetric and triangle-free pseudorandom graph G_n, there is a canonical G_n-realizable distribution on a domain of size n which is at distance $1/2$ from any i.i.d. mix. However, if a distribution on a fixed domain $\mathcal{D}$ is realizable on a family of pseudorandom graphs with $\frac{\lambda}{d} \to 0$, then it must be an i.i.d. mix.

Finally, (3) is our most technically involved result (which in the limit form subsumes result (2); however, the quantitative bounds here are much weaker). It shows that if the second normalized eigenvalue for a sequence $\{G_n\}$ tends to zero, then it is sufficient to assume just that ρ is $\vec{G}_n$-realizable for some *arbitrary orientation* of each G_n. In other words,

$(X_i, X_j) \sim \rho$ for some arbitrary direction of each edge $(i,j) \in E$, and this already implies that ρ is an i.i.d. mix. This is motivated by a version of de Finetti's theorem which holds under the weaker hypothesis that for every n, the marginal distributions induced by all *strictly increasing* sequences $(i_1, i_2, \ldots, i_n)$ are identical. (See [12] for a short proof using ergodic theory, or [21] for a finite version of this statement.) In particular, [21] proves that if $(X_i, X_j) \sim \rho$ for every $i < j$ for sufficiently many random variables, then ρ must be close to an i.i.d. mix. Our result implies that if $(X_i, X_j) \sim \rho$ for at least one (arbitrary) ordering of each pair $i, j \in [n]$, i.e. ρ is $\vec{T}$-realizable for some *tournament* $\vec{T}$, then ρ must be close to an i.i.d. mix. More generally, we prove this for orientations of pseudorandom graphs. Apart from properties of pseudorandom graphs, the proof of this result relies on the sparse regularity lemma, together with a counting argument adapted from [21].

1.1 Characterization of Realizable k-variate Distributions

We state the following (formal) strengthening of the Diaconis-Freedman theorem that illustrates the relationship between the notions of realizability and exchangeability, which are formally different but in some contexts closely related.

▶ **Proposition 1.** *Fix a k-variate distribution ρ. Suppose there are random variables $X_1, X_2, \ldots, X_n$ such that the marginal distribution of $(X_{i_1}, X_{i_2} \ldots, X_{i_k})$ equals ρ for every k-tuple $(i_1, i_2, \ldots, i_k)$ of distinct indices in $[n]$. Then ρ is $\frac{k(k-1)}{2n}$-close in variation distance to an i.i.d. mix. Therefore, if this property holds for all n, then ρ is an i.i.d. mix.*

Note that we assume only that every k distinct indices realizes the same distribution, rather than a full exchangeability of the sequence. Nonetheless, the result follows easily from the Diaconis-Freedman theorem as follows: Take a random permutation π of $[n]$ and consider the sequence $V_1, V_2, \ldots, V_n$, where $V_i = X_{\pi(i)}$ for $i \in [n]$. Then $V_1, V_2, \ldots, V_n$ is an exchangeable sequence that also realizes ρ on all k distinct indices. Therefore, ρ is $\frac{k(k-1)}{2n}$-close to an i.i.d. mix by the Diaconis-Freedman theorem. In the following, we give a short self-contained proof to illustrate the techniques that will be useful in the sequel. We will use the following standard fact from information theory.

▶ **Proposition 2** (Data-processing inequality for total variation distance)**.** *Let A, B, X be random variables such that X is independent of (A, B). Then for all functions h:*

$$d_{TV}(h(A,X), h(B,X)) \leq \max_x d_{TV}(h(A,x), h(B,x)) \leq d_{TV}(A,B)$$

Proof of Proposition 1. Pick uniformly a random k-tuple of distinct indices $(i_1, \ldots, i_k)$, and let $(Y_1, \ldots, Y_k) = (X_{i_1}, \ldots, X_{i_k})$. By the realizability property, $(Y_1, \ldots, Y_k) \sim \rho$. Define another distribution as follows: pick uniformly random and *independent* indices $j_1, \ldots, j_k \in [n]$ and define $(Z_1, \ldots, Z_k) = (X_{j_1}, \ldots, X_{j_k})$. Let $(Z_1, \ldots, Z_k) \sim \mu$.

We show that the two distributions are close to each other in variation distance. Note that the two distributions are obtained by choosing k indices by sampling with or without replacement, respectively, and then followed by applying the same (random) function $f(\ell) = X_\ell$ to each selected index ℓ. Since the sampling process was chosen independently of $X_1, X_2, \ldots, X_n$, the data-processing inequality implies that the variation distance between μ and ρ is at most that between sampling k times from $[n]$ with and without replacement. It is easy to see that this variation distance is upper-bounded by the probability that there are at least two equal indices among k independent samples from $[n]$; by the union bound, this probability is at most $\binom{k}{2}\frac{1}{n} = \frac{k(k-1)}{2n}$. ◀

2 Realizability on Graphs: Basic Properties

A distribution ρ over $\mathcal{D} \times \mathcal{D}$ is *$\vec{G}$-realizable* for a directed graph $\vec{G} = (V, A)$, if there is a joint distribution of random variables $(X_v : v \in V)$ such that $(X_i, X_j) \sim \rho$, for all arcs $(i, j) \in A$. Extend this definition to undirected graphs G by requiring that ρ is symmetric and that $(X_i, X_j) \sim \rho$, for all edges $\{i, j\} \in E$. We develop a few basic properties of G-realizability.

2.1 Realizability and Homomorphisms

A *homomorphism* from G to H is a mapping $f : V(G) \to V(H)$ such that $\{i, j\} \in E(G) \Rightarrow \{f(i), f(j)\} \in E(H)$ (for undirected graphs); and $(i, j) \in A(G) \Rightarrow (f(i), f(j)) \in A(H)$ (for directed graphs), for all i, j. We write $G \to H$ if such a homomorphism exists. We observe the following simple connection between realizability and homomorphisms.

▶ **Lemma 3.** *If ρ is H-realizable and $G \to H$ (directed or undirected), then ρ is G-realizable.*

Proof. Let $(Y_v : v \in V(H))$ be a collection of random variables such that for each edge $(i, j) \in E(H)$, (Y_i, Y_j) is distributed according to ρ. Let $f : G \to H$ be a homomorphism. Then we define a collection of random variables $(X_u : u \in V(G))$ such that $X_u = Y_{f(u)}$. For each edge $(i, j) \in E(G)$, we have $(f(i), f(j) \in E(H)$ and hence $(X_i, X_j) = (Y_{f(i)}, Y_{f(j)})$ is distributed according to ρ. This proves that ρ is G-realizable. ◀

It is well known (e.g., Proposition 3.5 in [17]) that for every graph G (directed or undirected), there is a unique minimal graph H such that $G \to H$ and $H \to G$. We call H the *core* of G and denote it as $C(G)$.

▶ **Corollary 4.** *A distribution ρ is G-realizable if and only if ρ is $C(G)$-realizable.*

In other words, it is only the core $C(G)$ that determines what distributions are G-realizable. For example, core of each bipartite graph is K_2 (a single edge), where every symmetric distribution is realizable. Containing a clique K_q is equivalent to the existence of a homomorphism $K_q \to G$. In particular, having a clique number $\omega(G) = q$ means that every G-realizable distribution satisfies the assumptions of Proposition 1 with $k = 2, n = q$, and is at distance at most $1/q$ from an i.i.d. mix. Conversely, having chromatic number $\chi(G) = k$ is equivalent to $G \to K_k$. This implies that any distribution realizable on K_k is also realizable on G. In particular, sampling twice without replacement from $[k]$ is realizable on G, which is at distance $1/k$ from an i.i.d. mix.

2.2 Symmetric (Edge/Arc-transitive) Graphs

Next, we consider the class of symmetric graphs, which possess a canonical realizable distribution — one corresponding to the adjacency matrix of the graph itself. We prove that in some sense all distributions realizable on symmetric graphs arise in this fashion.

▶ **Definition 5.** A directed graph $G = (V, E)$ is arc-transitive, if for each pair of directed edges $(u, v), (w, z) \in E$, there is an automorphism $\pi : V \to V$ such that $\pi(u) = w$ and $\pi(v) = z$. An undirected graph $G = (V, E)$ is symmetric, if for each pair of edges $\{u, v\}, \{w, z\} \in E$ and each pairing $u - w, v - z$ of their endpoints, there is an automorphism $\phi : V \to V$ such that $\pi(u) = w$ and $\pi(v) = z$.

We now characterize all G-realizable distributions for symmetric graphs. In short, the distributions are obtained by labeling the vertices of G arbitrarily with values in $\mathcal{D}$ and taking the labeling of a uniformly random edge. We need a preliminary fact.

▶ **Lemma 6.** *For an arc-transitive directed graph $\vec{G} = (V, A)$ and any arc $(u, v) \in A$, if π is a uniformly random automorphism of $\vec{G}$ then $(\pi(u), \pi(v))$ is a uniformly random arc in E.*

Proof. Let $\mathcal{A}$ denote the set of automorphisms of $\vec{G}$. Fix $(u, v) \in A$. For every arc $(w, z) \in A$ (possibly equal to (u, v)), define $\mathcal{A}_{wz} = \{\pi \in \mathcal{A} : \pi(u) = w, \pi(v) = z\}$. Fix an automorphism $\pi_1 \in \mathcal{A}_{wz}$ (which exists by arc-transitivity) and define $\phi : \mathcal{A} \to \mathcal{A}$ by $\phi(\pi) = \pi_1 \circ \pi$. We claim that ϕ is a bijection between $\mathcal{A}_{uv}$ and $\mathcal{A}_{wz}$: For each $\pi_0 \in \mathcal{A}_{uv}$, $(\phi(\pi_0))(u) = \pi_1(\pi_0(u)) = \pi_1(u) = w$ and $(\phi(\pi_0))(v) = \pi_1(\pi_0(v)) = z$. Similarly, $\phi^{-1}(\pi) = \pi_1^{-1} \circ \pi$ is the inverse of ϕ and maps $\mathcal{A}_{wz}$ to $\mathcal{A}_{uv}$. Therefore, $|\mathcal{A}_{uv}| = |\mathcal{A}_{wz}|$ and this holds for every $(w, z) \in A$. Thus a random automorphism is equally likely to map (u, v) to any other arc. ◀

▶ **Theorem 7.** *Let $G = (V, E)$ be a symmetric (undirected) or arc-transitive (directed) graph. Then ρ is a G-realizable distribution on $\mathcal{D} \times \mathcal{D}$ if and only if ρ is a convex combination of distributions ρ_f, where ρ_f for $f : V \to \mathcal{D}$ is defined as follows: (X, Y) is distributed according to ρ_f, if $(X, Y) = (f(u), f(v))$ where (u, v) is a uniformly random arc/edge of E (randomly ordered in the undirected case).*

Proof. First, if G is undirected, let us replace it by its bidirected version $\vec{G}$. Observe that a uniformly random arc (u, v) in $\vec{G}$ corresponds to a random ordering of a uniformly random edge in G, therefore our definition of ρ_f for directed/undirected graphs is consistent with this reduction to the directed case.

We prove that the distribution ρ_f is $\vec{G}$-realizable for every $f : V \to \mathcal{D}$. We define a random variable X_v for each vertex $v \in V$: $X_v = f(\pi(v))$ where π is a uniformly random automorphism of G. By Lemma 6, $(\pi(u), \pi(v))$ for any fixed edge $(u, v) \in E$ is uniformly distributed over all edges in E (uniformly over both orderings in the undirected case). Therefore, $(X_u, X_v) = (f(\pi(u)), f(\pi(v)))$ is distributed according to ρ_f.

Conversely, assume that ρ is $\vec{G}$-realizable and let $\{X_v : v \in V\}$ be a collection of random variables realizing ρ on each edge $(u, v) \in E$. Consider a pair of random variables (Y, Z) generated by $(Y, Z) = (X_u, X_v)$ where (u, v) is a uniformly random edge in E (randomly ordered in the undirected case). Since conditioned on (u, v), the distribution of (X_u, X_v) is ρ, the distribution of (Y, Z) is also ρ. We claim that the distribution of (Y, Z) is a convex combination of distributions ρ_f as in the statement above. To see this, consider some assignment of values $X_v = f(v)$ of nonzero probability. Conditioned on $X_v = f(v)\ \forall v \in V$, we have $(Y, Z) = (f(u), f(v))$ where (u, v) is a random edge — i.e., (Y, Z) is distributed according to ρ_f. Therefore, ρ is a convex combination of such distributions. ◀

In particular, the distribution ρ_{Id} on $\mathcal{D} = V$, obtained by taking $(X, Y) =$ *uniformly random edge of G* is G-realizable for any symmetric graph. We call ρ_{Id} the *canonical G-realizable distribution*. Using a classical theorem of Motzkin and Straus [16], we can show that the exact distance of ρ_{Id} from an i.i.d. mix is determined by the clique number of G.

▶ **Theorem 8.** *For an undirected symmetric graph G, the canonical G-realizable distribution ρ_{Id} is at variation distance exactly $1/\omega(G)$ from an i.i.d. mix, where $\omega(G)$ is the maximum size of a clique in G. This is the maximum distance from an i.i.d. mix among all G-realizable distributions.*

Proof. Let A be the adjacency matrix of G. The Motzkin-Straus theorem [16] states that

$$\max_{\|p\|_1, p \geq 0} p^T A p = 1 - \frac{1}{\omega(G)}.$$

Therefore, for any probability distribution p on V, we have $\sum_{a,b \in V} p_a p_b A_{ab} \leq 1 - \frac{1}{\omega(G)}$. By taking convex combinations, for any i.i.d. mix $\mu_{ab} = \sum \alpha_i p_a^i p_b^i$, we still have $\sum_{a,b \in V} \mu_{ab} A_{ab} \leq$

$1 - \frac{1}{\omega(G)}$. In other words, μ_{ab} has at most $1 - \frac{1}{\omega(G)}$ probability mass on the edges of G. However, the canonical G-realizable distribution ρ_{Id} is supported on the edges of G. Therefore, it is at distance at least $\frac{1}{\omega(G)}$ from μ. As we discussed above, every G-realizable distribution ρ must be at distance at most $\frac{1}{\omega(G)}$ from an i.i.d. mix, due to Proposition 1. Therefore, ρ_{Id} has distance exactly $\frac{1}{\omega(G)}$ from an i.i.d. mix. ◀

▶ **Example 9.** For a cycle C_n, the canonical C_n-realizable distribution ρ is defined as follows: $\rho(i, i+1 \bmod n) = \rho(i, i-1 \bmod n) = \frac{1}{2n}$; this is at variation distance $1/2$ from an i.i.d. mix.

3 Realizability Based on Spectral Properties

In this section we consider undirected graphs. We showed in Theorem 7 that each fixed symmetric graph G possesses a rather rich collection of G-realizable distributions, similar to the structure of the graph itself. However, perhaps a more interesting question is: What fixed distributions are G_n-realizable for a family of graphs $\{G_n\}$ of growing size? Here we do not have many non-trivial examples, other than those where all the graphs G_n map homomorphically to a fixed symmetric graph H, and then we can realize all the H-realizable distributions. Similar to de Finetti's theorem, one can ask — what are the families of graphs that admit only i.i.d. mix distributions to be realized?

▶ **Example 10.** Consider any family of d-regular graphs $\{G_n\}$, for example d-regular expanders (even Ramanujan graphs). These graphs are $(d+1)$-colorable, therefore any K_{d+1}-realizable distribution is also realizable on each G_n. In particular, sampling from $[d+1]$ without replacement is realizable and at a fixed distance $\frac{1}{d+1}$ from an i.i.d. mix; i.e this family does not force a realizable distribution to be an i.i.d. mix.

On the other hand, even though this family may not contain any clique K_{d+1}, even any triangles, it seems to force realizable distributions to be quite close to an i.i.d. mix. This motivates us to investigate the relationship of realizability and spectral properties of graphs.

▶ **Definition 11.** The eigenvalues of a graph G are the eigenvalues of its adjacency matrix $A(G)$, $(A(G))_{ij} = 1$ if $\{i, j\} \in E(G)$ and 0 otherwise.

It is known that the eigenvalues are all real and contained in $[-\Delta, \Delta]$ where Δ is the maximum degree in G. We order the eigenvalues in a descending order and label them $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. By the trace formula, we have $\sum_{i=1}^{n} \lambda_i = 0$. The gap between λ_1 and λ_2 is related to expansion properties of G, while the minimum (most negative) eigenvalue λ_n is related to how close G is to a bipartite graph. For a bipartite graph, we have $\lambda_n = -\lambda_1$. Graphs where λ_n/λ_1 is close to zero are those where "MaxCutGain" is small: for any two disjoint sets $A, B \subset V$, the number of edges between A and B is not significantly larger than the number of edges inside A and B (see [20]). Thus these graphs can be considered "far from bipartite". In the following, we prove that this property imposes a natural condition on what distributions are G-realizable.

3.1 Graphs Without Large Negative Eigenvalues

We begin with a result that is proved using a standard spectral argument.

▶ **Lemma 12.** *Let ρ be a G-realizable distribution on $\mathcal{D} \times \mathcal{D}$ for a graph $G = (V, E)$, and let μ be the marginal distribution of X in $(X, Y) \sim \rho$. Then for any function $\phi : \mathcal{D} \to \mathbb{R}$,*

$$\mathop{\mathbb{E}}_{(X,Y)\sim\rho}[\phi(X)\phi(Y)] \geq \frac{\lambda_n(G)}{\lambda_1(G)} \mathop{\mathbb{E}}_{X\sim\mu}[\phi^2(X)].$$

Proof. We can assume that G has no isolated vertices (which add only zero eigenvalues to the spectrum). Let $(X_v : v \in V)$ be a collection of random variables realizing ρ on G. Since the distribution of (X_i, X_j) is ρ for each edge $(i,j) \in E$, we have

$$\mathop{\mathbb{E}}_{(X,Y)\sim\rho}[\phi(X)\phi(Y)] = \frac{1}{|E|}\sum_{\{i,j\}\in E}\mathop{\mathbb{E}}_{(X_v:v\in V)}[\phi(X_i)\phi(X_j)] = \frac{1}{|E|}\mathop{\mathbb{E}}_{(X_v:v\in V)}[\sum_{\{i,j\}\in E}\phi(X_i)\phi(X_j)].$$

Let A be the adjacency matrix of G. The sum inside the expectation is a quadratic form which can be lower-bounded using the minimum eigenvalue of A:

$$\sum_{\{i,j\}\in E}\phi(X_i)\phi(X_j) = \frac{1}{2}\sum_{i,j\in V}\phi(X_i)A_{ij}\phi(X_j) \geq \frac{1}{2}\lambda_n\sum_{i\in V}\phi^2(X_i).$$

Let us take the expectation over the distribution of $(X_v : v \in V)$. Using the fact that each $v \in V$ is in some edge and hence the marginal distribution of X_v is μ, we obtain

$$\mathop{\mathbb{E}}_{(X,Y)\sim\rho}[\phi(X)\phi(Y)] \geq \frac{\lambda_n}{2|E|}\mathop{\mathbb{E}}_{(X_v:v\in V)}[\sum_{i\in V}\phi^2(X_i)] = \frac{|V|\lambda_n}{2|E|}\mathop{\mathbb{E}}_{X\sim\mu}[\phi^2(X)] \leq \frac{\lambda_n}{\lambda_1}\mathop{\mathbb{E}}_{X\sim\mu}[\phi^2(X)],$$

since $2|E| = \mathbf{1}^T A\mathbf{1} \leq \lambda_1\|\mathbf{1}\|^2 = \lambda_1|V|$. This proves the lemma. ◀

Using the above lemma, we can perturb the distribution ρ and obtain a probability matrix which is positive semidefinite.

▶ **Theorem 13.** *If ρ is a G-realizable distribution, then ρ is at distance at most $|\frac{\lambda_n(G)}{\lambda_1(G)}|$ from a distribution ρ' whose probability matrix $\rho'_{ab} = \Pr_{(X,Y)\sim\rho'}[X = a, Y = b]$ is positive semidefinite.*

Proof. Let ρ be a G-realizable distribution on $\mathcal{D}\times\mathcal{D}$ and let μ be its marginal on the first variable, as in Theorem 12. Let $\delta_{ab} = 1$ if $a = b$ and 0 otherwise. We define

$$\rho'_{ab} = \frac{\rho_{ab} + |\frac{\lambda_n}{\lambda_1}|\mu_a\delta_{ab}}{1 + |\frac{\lambda_n}{\lambda_1}|}.$$

It can be checked easily that $\rho'_{ab} \geq 0$ and $\sum_{a,b\in\mathcal{D}}\rho'_{ab} = 1$, so this is a probability distribution. The total variantion distance between ρ and ρ' is

$$\begin{aligned}\frac{1}{2}\|\rho-\rho'\|_1 &= \frac{1}{2}\sum_{a,b\in\mathcal{D}}|\rho_{ab}-\rho'_{ab}| = \frac{1}{2(1+|\frac{\lambda_n}{\lambda_1}|)}\sum_{a,b\in\mathcal{D}}|(1+\left|\frac{\lambda_n}{\lambda_1}\right|)\rho_{ab} - (\rho_{ab}+\left|\frac{\lambda_n}{\lambda_1}\right|\mu_a\delta_{ab})| \\ &= \frac{|\frac{\lambda_n}{\lambda_1}|}{2(1+|\frac{\lambda_n}{\lambda_1}|)}\sum_{a,b\in\mathcal{D}}|\rho_{ab}-\mu_a\delta_{ab}| \leq \frac{|\frac{\lambda_n}{\lambda_1}|}{1+|\frac{\lambda_n}{\lambda_1}|}.\end{aligned}$$

We claim that ρ' is a positive-semidefinite matrix: For any $\phi : \mathcal{D} \to \mathbb{R}$, we have

$$\sum_{a,b\in\mathcal{D}}\rho'_{ab}\phi(a)\phi(b) = \frac{1}{1+|\frac{\lambda_n}{\lambda_1}|}\left(\sum_{a,b\in\mathcal{D}}\rho_{ab}\phi(a)\phi(b) + \left|\frac{\lambda_n}{\lambda_1}\right|\sum_{a\in\mathcal{D}}\mu_a\phi^2(a)\right).$$

By Theorem 12, $\sum_{a,b\in\mathcal{D}}\rho_{ab}\phi(a)\phi(b) \geq \frac{\lambda_n}{\lambda_1}\sum_{a\in\mathcal{D}}\mu_a\phi^2(a)$ (which is a negative number), so we obtain $\sum_{a,b\in\mathcal{D}}\rho'_{ab}\phi(a)\phi(b) \geq 0$. ◀

This has the following corollary which is one of the results claimed in the introduction.

▶ **Corollary 14.** *Let ρ be a distribution over $\mathcal{D} \times \mathcal{D}$ such that for an arbitrarily large n, ρ is G_n-realizable for a graph G_n such that $|V(G_n)| = n$ and*

$$\lim_{n\to\infty} \frac{\lambda_n(G_n)}{\lambda_1(G_n)} = 0,$$

then $\rho_{ab} = \Pr_{(X,Y)\sim\rho}[X = a, Y = b]$ is a positive-semidefinite matrix.

Therefore, distributions ρ realizable on a sequence of graphs with normalized minimum eigenvalue tending to 0 must be positive semidefinite. Since $\sum_{a,b\in\mathcal{D}} \rho_{ab} = 1$ and $\rho_{ab} \geq 0$, this seems close to the condition of being an i.i.d. mix: $\rho_{ab} = \sum_i q_i p_i(a) p_i(b)$ for distributions $\sum_i q_i = 1$ and $\sum_{a\in\mathcal{D}} p_i(a) = 1$. However, these two conditions have been extensively studied and they are not equivalent.

3.2 Completely Positive and Doubly Nonnegative Matrices

▶ **Definition 15.** A matrix $A \in \mathbb{R}^{n\times n}$ is *completely positive* if $\forall i, j; A_{ij} = \mathbf{v}_i \cdot \mathbf{v}_j$ for nonnegative vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in \mathbb{R}^n_+$. A matrix $A \in \mathbb{R}^{n\times n}$ is *doubly nonnegative* if A is positive semidefinite and $\forall i, j; A_{ij} \geq 0$.

In our setting, completely positive matrices correspond to i.i.d. mixes (up to normalization), while doubly nonnegative matrices correspond to the distributions arising in Corollary 14. Clearly, every completely positive matrix is doubly nonnegative, but the opposite is not true. Nonetheless, the smallest known counterexamples are 5×5 matrices, and in fact for matrices up to 4×4 the two conditions are equivalent.

▶ **Theorem 16** ([15])**.** *A matrix in $\mathbb{R}^{4\times4}$ is completely positive if and only if it is doubly nonnegative.*

Therefore, we obtain the following corollary for domains of size up to 4 (in particular, Boolean random variables).

▶ **Corollary 17.** *If ρ is a distribution on $\mathcal{D} \times \mathcal{D}$, $|\mathcal{D}| \leq 4$, and ρ is G_n-realizable for an arbitrarily large n on graphs G_n such that $|V(G_n)| = n$ and*

$$\lim_{n\to\infty} \frac{\lambda_n(G_n)}{\lambda_1(G_n)} = 0,$$

then ρ is an i.i.d. mix.

Proof. By Corollary 14, $\rho_{ab} \in \mathbb{R}^{\mathcal{D}\times\mathcal{D}}$ is a doubly nonnegative matrix, and hence for $|\mathcal{D}| \leq 4$ it is completely positive: $\rho_{ab} = \mathbf{v}_a \cdot \mathbf{v}_b$ for vectors $\mathbf{v}_a \geq 0$. For each coordinate i, let $q_i = \sum_{a\in\mathcal{D}} v_{ai}$ and define $p_i(a) = v_{ai}/q_i$ (we can assume $q_i > 0$, otherwise we remove that coordinate). We have $p_i(a) \geq 0$ and $\sum_{a\in\mathcal{D}} p_i(a) = \sum_{a\in\mathcal{D}} v_{ai}/q_i = 1$, so each $p_i(a)$ is a distribution on $\mathcal{D}$ and we have $\rho_{ab} = \sum_i q_i p_i(a) p_i(b)$. Since $\sum_{a,b\in\mathcal{D}} \rho_{ab} = 1$, it is easy to verify that $\sum q_i = 1$ as well. ◀

3.3 Realizability on Pseudorandom Graphs

Next, we show that if we assume that all eigenvalues except for one are close to 0, then G-realizable distributions must be close to an i.i.d. mix. We follow the exposition on pseudorandom graphs in [13].

▶ **Definition 18.** G is an (n, d, λ)-graph if G is a d-regular graph on n vertices such that all eigenvalues except for the largest one are bounded by λ in absolute value.

Note that the largest eigenvalue itself equals d. The definition above implies pseudorandom properties whenever λ/d is small. This is a rather strong definition of pseudorandomness; however, most of the known constructions of pseudorandom graphs fall in this category.

▶ **Definition 19.** For an undirected graph G and two sets of vertices $S, T \subseteq V(G)$ (not necessarily disjoint), we define $e(S,T) = |\{u \in S, v \in T : \{u,v\} \in E(G)\}|$ to denote the number of edges between S and T. For a directed graph $\vec{G}$, we define $\vec{e}(S,T) = |\{u \in S, v \in T : (u,v) \in E(G)\}|$ to denote the number of arcs from S to T.

Note that in the undirected case, each edge inside $S \cap T$ is counted twice in $e(S,T)$. In the directed case, each arc inside $S \cap T$ is counted once in $\vec{e}(S,T)$. (This is consistent with the view that an undirected graph can be viewed as a directed graph by replacing each undirected edge with the two arcs representing possible orientations of that edge.)

Next, we formulate the well-known *expander mixing lemma* (for undirected graphs).

▶ **Proposition 20** (Expander Mixing Lemma [13])**.** *Let G be an (n,d,λ)-graph. Then for any $S, T \subseteq V(G)$,*

$$\left| e(S,T) - \frac{d}{n}|S| \cdot |T| \right| \leq \frac{\lambda}{n}\sqrt{|S|(n-|S|)|T|(n-|T|)} \leq \lambda\sqrt{|S||T|}$$

In particular, this implies that (n,d,λ)-graphs for small values of $\frac{\lambda}{d}$ are good expanders (by taking $T = V \setminus S$).

▶ **Theorem 21.** *Let G be an (n,d,λ)-graph. If a distribution ρ is G-realizable, then ρ is $(\frac{\lambda}{d} \cdot \sqrt{|\mathcal{D}|})$-close in variation distance to an i.i.d. mix.*

Proof. Let ρ be a G-realizable distribution and let $\mathbf{X} = (X_1, X_2, \ldots, X_n)$ certify the G-realizability of ρ. Let $\vec{G}$ denote the directed graph obtained by replacing each undirected edge by two arcs in opposite directions. Choose an edge $\vec{e} = (i,j)$ uniformly from $E(\vec{G})$ which consists of nd arcs. By G-realizability, it follows that the distribution of (X_i, X_j) also equals ρ. As before, conditioned on $\mathbf{X} = \mathbf{x}$, for a uniformly chosen vertex $i \in V(G)$ the distribution of X_i is the empirical distribution $p_{\mathbf{x}}$. So the distribution μ of (X_i, X_j) when (i,j) is chosen uniformly from $[n]^2$ is an i.i.d. mix, a convex combination of products of such empirical distributions.

We now show that $d_{TV}(\mu, \rho)$ is small. By the data-processing inequality, it suffices to show that this holds when conditioned on $\mathbf{X} = \mathbf{x}$, for every $\mathbf{x}$. Let $\phi = (\phi_{ab})$ (respectively, ψ) be the probability distribution of (x_i, x_j) when when (i,j) is uniformly random in $[n]^2$ (respectively, (i,j) is uniformly random in $E(\vec{G})$). Let V_a denote the set of vertices labeled a, for each $a \in \mathcal{D}$. Note that the label sets partition V. Then $\phi_{ab} = |V_a| \cdot |V_b|/n^2$. On the other hand, it can be checked for both the cases $a = b$ and $a \neq b$ that $\psi_{ab} = e(V_a, V_b)/(nd)$.

Let $Q = \{(a,b) \in \mathcal{D} : \phi(a,b) < \psi(a,b)\}$ so that $d_{TV}(\phi, \psi) \leq \sum_{(a,b)\in Q}(\psi_{ab} - \phi_{ab})$. Fix $a \in \mathcal{D}$ and let $W_a = \bigcup_{b:(a,b)\in Q} V_b$. Observe that $\sum_{b:(a,b)\in Q} e(V_a, V_b) = e(V_a, W_a)$, since the V_a's are pairwise disjoint. Similarly $\sum_{b:(a,b)\in Q} |V_b| = |W_a|$. By the Expander Mixing Lemma:

$$\sum_{b:(a,b)\in Q} (\psi_{ab} - \phi_{ab}) = \frac{1}{nd}\Big(e(V_a, W_a) - \frac{d}{n}|V_a| \cdot |W_a|\Big) \leq \frac{\lambda}{nd}\sqrt{|V_a| \cdot |W_a|} \leq \frac{\lambda}{d}\sqrt{\frac{|V_a|}{n}} \quad (1)$$

By the Cauchy-Schwarz inequality, $\sum_{a\in\mathcal{D}} \sqrt{|V_a|/n} \leq \sqrt{|\mathcal{D}|} \cdot \sqrt{\sum_{a\in\mathcal{D}} |V_a|/n} = \sqrt{|\mathcal{D}|}$, because the label sets partition V. Using this bound, we conclude the proof using eq. (1) as follows:

$$d_{TV}(\phi, \psi) \leq \sum_{a\in\mathcal{D}} \sum_{b:(a,b)\in Q} (\psi_{ab} - \phi_{ab}) \leq \frac{\lambda}{d} \sum_{a\in\mathcal{D}} \sqrt{\frac{|V_a|}{n}} \leq \frac{\lambda}{d} \cdot \sqrt{|\mathcal{D}|}$$ ◀

▶ **Corollary 22.** *If for arbitrarily large n, ρ is G_n-realizable on an $(n, d(n), \lambda(n))$-graph G_n such that*

$$\lim_{n\to\infty} \frac{\lambda(n)}{d(n)} = 0$$

then ρ is an i.i.d. mix.

In other words, the only distributions realizable on a family of pseudorandom graphs with $\frac{\lambda}{d} \to 0$ are i.i.d. mixes. However, it is not true that a distribution realizable on a single pseudorandom graph with very good parameters must be close to an i.i.d. mix. In contrast to the Diaconis-Freeman theorem, some dependence on $|\mathcal{D}|$ seem to be necessary in Theorem 21.

▶ **Example 23.** Consider a *symmetric* (n, d, λ)-graph G. It is known that there are such graphs with arbitrarily small λ/d (see e.g. [11, 9]; these graphs are described as edge-transitive but in fact can be seen to be symmetric in the stronger sense of Theorem 5 as well). In addition, these graphs have high girth [14, 10]; for us it is sufficient that they are triangle-free. Consider the canonical distribution $(X, Y) \sim \rho$ where $(X, Y) = (u, v)$ is a (randomly ordered) uniformly random edge of G. This is G-realizable by Theorem 7. By Theorem 8, this distribution has variation distance $1/\omega(G) = 1/2$ from any i.i.d. mix.

4 Orientations of Pseudorandom Graphs

Since we know that realizability on pseudorandom graphs implies being an i.i.d. mix, one can ask whether it is really necessary to require the distribution ρ to be symmetric to start with. Perhaps being realizable on a suitable directed graph already implies being an i.i.d. mix and in particular being symmetric?

First, we have the following simple lemma which shows that in non-trivial cases the marginals of ρ must be symmetric.

▶ **Lemma 24.** *Let ρ be $\vec{G}$-realizable for a non-bipartite directed graph $\vec{G}$. Then $\forall a \in \mathcal{D}$:*

$$\Pr_{(X,Y)\sim\rho}[X = a] = \Pr_{(X,Y)\sim\rho}[Y = a].$$

Proof. We claim that there is a vertex v that is a head and also a tail of some edge: $\exists u, w \in V; (u, v) \in A(\vec{G})$ and $(v, w) \in A(\vec{G})$. If not, the orientation of $\vec{G}$ is such that vertices can be divided into head-only and tail-only; but this means that $\vec{G}$ is bipartite.

Thus, we have random variables X_u, X_v, X_w such that $(X_u, X_v) \sim \rho$ and also $(X_v, X_w) \sim \rho$. So, $\Pr_{(X,Y)\sim\rho}[X = a] = \Pr_{(X,Y)\sim\rho}[Y = a] = \Pr[X_v = a]$. ◀

However, this does not mean that ρ must be symmetric. An example is a directed cycle $\vec{C}_n$ with vertices identified with $\mathbb{Z}_n$, where we can have the following distribution: $X_i = Z + i \pmod n$, where Z is uniformly random in $\mathbb{Z}_n$. Then the distribution on each directed edge is given by $(X_i, X_{i+1}) = (j, j + 1)$ for each $j \in \mathbb{Z}_n$ with probability $1/n$. This distribution has symmetric marginals but it is not symmetric.

Nevertheless, if ρ is $\vec{G}_n$-realizable on a sufficiently dense directed graph $\vec{G}_n$, it seems to force ρ to be symmetric. One example of this is the transitive tournament $A = \{(i, j) : i, j \in [n], i < j\}$: If (X_i, X_j) has the same distribution for each $i < j$, then the distribution must be close to an i.i.d. mix. (This was proved implicitly by Trotter and Winkler [21].) Is being realizable on an arbitrary tournament sufficient to conclude that ρ must be close to an i.i.d. mix? In this section, we consider an even more general question: If ρ is realizable on some arbitrary orientation of a pseudorandom graph, does ρ have to be close to an i.i.d. mix? We prove that the answer is yes.

▶ **Theorem 25.** *Let ρ be a distribution such that for an arbitrarily large n, ρ is $\vec{G}_n$-realizable for some orientation $\vec{G}_n$ of an $(n, d(n), \lambda(n))$-graph and*

$$\lim_{n\to\infty} \frac{\lambda(n)}{d(n)} = 0.$$

Then ρ is an i.i.d. mix.

4.1 Directed Sparse Regularity Lemma

The main tool that we use here is a directed version of the *sparse regularity lemma*. The sparse regularity lemma of Kohayakawa and Rödl says roughly that any subgraph of a "well-behaved graph" of a certain density p, for example a pseudorandom graph, can be partitioned in such a way that most pairs of parts are regular with an error proportional to p. In addition, we need a directed version of this lemma. We follow the exposition in [7], Section 2.1.

▶ **Definition 26.** For a directed graph $\vec{G} = (V, A)$ and a parameter $p > 0$, we define the oriented p-density for a pair of sets $U, W \subseteq V$ as

$$d_p(U, W) = \frac{2\vec{e}(U, W)}{p|U||W|}.$$

▶ **Definition 27.** A directed graph $\vec{G} = (V, A)$ is (η, D, p)-bounded if, for any pair of disjoint sets $U, W \subseteq V$ with $|U|, |W| \geq \eta|V|$, we have $d_p(U, W) \leq D$.

▶ **Definition 28.** A pair of disjoint sets $U, W \subseteq V$ is (ϵ, p)-regular if for all $U' \subseteq U, |U'| \geq \epsilon|U|$ and $W' \subseteq W, |W'| \geq \epsilon|W|$, we have

$$|d_p(U', W') - d_p(U, W)| < \epsilon.$$

A partition $\mathcal{P} = \{V_0, V_1, \ldots, V_k\}$ of V is (ϵ, k, p)-regular if $|V_0| \leq \epsilon|V|$, $|V_1| = |V_2| = \ldots = |V_k|$ and for more than $(1 - \epsilon)\binom{k}{2}$ pairs $\{i, j\} \subseteq [k], i \neq j$, we have that (V_i, V_j) and (V_j, V_i) are both (ϵ, p)-regular.

▶ **Lemma 29** (directed sparse regularity lemma, [7])**.** *For any real $\epsilon > 0$, $D > 1$ and integer $k_0 \geq 1$ there exists $\eta > 0$ and $K \geq k_0$ such that for every $0 < p \leq 1$, every (η, D, p)-bounded directed graph $\vec{G}$ admits an (ϵ, k, p)-regular partition for some $k_0 \leq k \leq K$.*

4.2 Application of the Sparse Regularity Lemma

Starting with an arbitrary orientation of a pseudorandom graph $\vec{G}$, we use the regularity lemma to identify a certain bipartite subgraph (A, B) of $\vec{G}$ where the orientation behaves also in a pseudorandom way, with a significant density β in one direction.

▶ **Lemma 30.** *For every $\epsilon \in (0, \frac{1}{2})$, there is $K \geq 2$ and $\gamma > 0$ such that given any orientation of an (n, d, λ)-graph $\vec{G}$ with $\lambda \leq \gamma d$, there are disjoint sets $A, B \subseteq V$, $|A| = |B| \geq \frac{n}{2K}$ and $\beta = \frac{\vec{e}(A,B)}{|A||B|} \geq \frac{d}{4n}$ such that*

$$\forall A' \subseteq A, B' \subseteq B; \left|\vec{e}(A', B') - \beta|A'||B'|\right| < 8\epsilon\beta|A||B|.$$

Proof. Fix $D = 4$ and $k_0 = 1$. Given $\epsilon \in (0,1)$, let $\eta > 0$ and $K \geq 1$ be the parameters given by Lemma 29. We set $\gamma = \min\{\eta, \frac{\epsilon}{4K}\}$. Then consider any orientation $\vec{G}$ of an (n,d,λ)-graph such that $\lambda \leq \gamma d$. We set $p = \frac{d}{n}$.

First we verify the condition of (η, D, p)-boundedness. Applying Proposition 20 to the undirected version of $\vec{G}$, we have

$$\left| e(S,T) - \frac{d}{n}|S||T| \right| \leq \lambda\sqrt{|S||T|} \leq \eta d\sqrt{|S||T|}$$

where $e(S,T) = \vec{e}(S,T) + \vec{e}(T,S)$ denotes the number of all edges between S and T (both directions). Our goal is to bound the normalized density $d_p(S,T) = \frac{2\vec{e}(S,T)}{p|S||T|}$. We have

$$\vec{e}(S,T) \leq \frac{d}{n}|S||T| + \eta d\sqrt{|S||T|} = p|S||T| + \eta pn\sqrt{|S||T|}.$$

For $|S|, |T| \geq \eta|V| = \eta n$, we obtain $\eta n \leq \sqrt{|S||T|}$, $\vec{e}(S,T) \leq 2p|S||T|$ and $d_p(S,T) = \frac{2\vec{e}(S,T)}{p|S||T|} \leq 4$. This verifies that $\vec{G}$ is (η, D, p)-bounded for $D = 4$. Consequently, Lemma 29 states that $\vec{G}$ admits an (ϵ, k, p)-regular partition $\mathcal{P} = \{V_0, V_1, \ldots, V_k\}$ where $k \leq K$.

Pick any (ϵ, p)-regular pair (V_i, V_j), $1 \leq i < j$. We have $|V_i| = |V_j| \geq \frac{1-\epsilon}{K}n \geq \frac{n}{2K}$. By Proposition 20 and the condition $\lambda \leq \gamma d \leq \frac{d}{4K}$, the number of undirected edges between V_i and V_j satisfies

$$\left| e(V_i,V_j) - \frac{d}{n}|V_i||V_j| \right| \leq \lambda\sqrt{|V_i||V_j|} \leq \frac{d}{4K}\sqrt{|V_i||V_j|} \leq \frac{d}{2n}|V_i||V_j|.$$

This implies that the number of undirected edges between V_i and V_j is $e(V_i,V_j) \geq \frac{d}{2n}|V_i||V_j|$. In at least one direction, assume from V_i to V_j, we get $\vec{e}(V_i,V_j) \geq \frac{d}{4n}|V_i||V_j|$. Then set $A = V_i$, $B = V_j$ and $\beta = \frac{\vec{e}(A,B)}{|A||B|}$. By the above we have $\beta \geq \frac{d}{4n}$. We claim that the conclusion of the lemma holds with these parameters.

We know that (A,B) is an (ϵ,p)-regular pair. This means that for any $A' \subseteq A$, $B' \subseteq B$ with $|A'| \geq \epsilon|A|, |B'| \geq \epsilon|B|$, we have

$$|d_p(A',B') - d_p(A,B)| < \epsilon.$$

Here, $d_p(A',B') = \frac{2\vec{e}(A',B')}{p|A'||B'|}$. So we can rewrite this bound as

$$\left| \frac{2\vec{e}(A',B')}{p|A'||B'|} - \frac{2\vec{e}(A,B)}{p|A||B|} \right| = \frac{2}{p}\left| \frac{\vec{e}(A',B')}{|A'||B'|} - \beta \right| < \epsilon.$$

Recalling that $p = \frac{d}{n} \leq 4\beta$,

$$\left| \vec{e}(A',B') - \beta|A'||B'| \right| < \frac{p}{2}\epsilon|A'||B'| \leq 2\beta\epsilon|A||B|.$$

We still have to handle the case where $|A'| < \epsilon|A|$ or $|B'| < \epsilon|B|$. Then applying Proposition 20 again (ignoring the regularity property of (A,B)), we obtain

$$\left| e(A',B') - \frac{d}{n}|A'||B'| \right| \leq \gamma d\sqrt{|A'||B'|}$$

and since $\gamma \leq \frac{\epsilon}{4K}$, $|A'||B'| \leq \epsilon|A||B|$ and $|A| = |B| \geq \frac{n}{2K}$, we get

$$\vec{e}(A',B') \leq \frac{d}{n}|A'||B'| + \gamma d\sqrt{|A'||B'|} \leq \frac{\epsilon d}{n}|A||B| + \frac{\epsilon d}{4K}\sqrt{|A||B|} \leq \frac{2\epsilon d}{n}|A||B|.$$

We have $\beta \geq \frac{d}{4n}$, thus we conclude that $\vec{e}(A',B') \leq 8\epsilon\beta|A||B|$. ◀

4.3 The Second Moment Argument

Once we have identified the regular pair (A, B), our goal is to prove that the realizability of ρ on this regular pair implies that ρ must be close to a symmetric distribution. We adapt an approach of Trotter and Winkler [21] for the case of a transitive tournament $\{(i,j) : i,j \in [n], i < j\}$. Roughly speaking, their argument is that if (X_i, X_j) has the same distribution for each $i < j$, then any particular value in $\mathcal{D}$ has a similar number of occurrences among $\{X_1, X_2, \ldots, X_{n/2}\}$ and in $\{X_{n/2+1}, \ldots, X_n\}$. Then, counting the pairs across the two blocks, the number of (a,b) pairs is similar to the number of (b,a) pairs for any $a, b \in \mathcal{D}$, which means that the distribution is close to symmetric. Technically, the proof involves a second moment computation and the Cauchy-Schwarz inequality. We show here that a similar argument still goes through in the setting of an (arbitrary) orientation of a pseudorandom graph.

The first part of the proof does not depend on the regularity of the directed pair (A, B). It uses only properties of the undirected pseudorandom graph G.

▶ **Lemma 31.** *Let $\xi \in (0, \frac{1}{2})$ and let ρ be a $\vec{G}$-realizable distribution for some orientation $\vec{G}$ of an (n,d,λ)-graph G (non-bipartite, without isolated vertices). Let $\{X_v : v \in V\}$ the random variables realizing ρ on $\vec{G}$. For every $a \in \mathcal{D}$ and $S \subseteq V$, define a random set*

$$S_a = \{v \in S : X_v = a\}.$$

Let $A, B \subseteq V$ be two disjoint sets such that $|A| = |B| \geq \frac{\lambda n}{\xi^2 d}$. Then for every $a, b \in \mathcal{D}$,

$$\Big|\mathbb{E}[|A_a||B_b| - |A_b||B_a|]\Big| \leq 4\xi\sqrt{\mu_a \mu_b}|A|^2$$

where $\mu_a = \Pr_{(X,Y)\sim\rho}[X = a]$.

Proof. We write the target quantity as follows:

$$\mathbb{E}[|A_a||B_b| - |A_b||B_a|] = \mathbb{E}[|A_a|(|B_b| - |A_b|) + (|A_a| - |B_a|)|A_b|].$$

By Cauchy-Schwarz,

$$\Big|\mathbb{E}[|A_a||B_b| - |A_b||B_a|]\Big| \leq \sqrt{\mathbb{E}[|A_a|^2]}\sqrt{\mathbb{E}[(|B_b| - |A_b|)^2]} + \sqrt{\mathbb{E}[|A_b|^2]}\sqrt{\mathbb{E}[(|A_a| - |B_a|)^2]}. \quad (2)$$

We estimate the second moments of $|A_a|$, $|A_b|$, etc. by relating $|A_a|^2$ to the number of edges inside A labeled (a,a), etc. This is possible, since G is pseudorandom and hence its edges are in some sense a good representation of all possible pairs.

Recall that A_a denotes the random subset of A whose random variables attain value a. Each vertex in G in contained in some directed edge; hence $\forall v \in V; \Pr[X_v = a] = \mu_a$ and $\mathbb{E}[|A_a|] = \mu_a|A|$ (see Lemma 24). Further, let us consider

$$e(S_a, T_a) = |\{u \in S, v \in T : \{u,v\} \in E(G), X_u = X_v = a\}|.$$

Since the distribution on each directed edge of $\vec{G}$ is ρ, the probability that $X_u = X_v = a$ for $\{u,v\} \in E(G)$ is ρ_{aa}. Note that the orientation does not matter here, because we are looking at the event of each endpoint having the same value. We obtain

$$\mathbb{E}[e(S_a, T_a)] = \rho_{aa} e(S,T).$$

On the other hand, $e(S_a, T_a)$ is equal to the number of edges between S_a and T_a. Since G is an (n,d,λ)-graph, Proposition 20 gives

$$\left|e(S_a, T_a) - \frac{d}{n}|S_a||T_a|\right| \leq \lambda\sqrt{|S_a||T_a|}.$$

We use this bound for the following choices of S and T:

- $e(A_a, A_a) \geq \frac{d}{n}|A_a|^2 - \lambda|A_a|$
- $e(B_a, B_a) \geq \frac{d}{n}|B_a|^2 - \lambda|B_a|$
- $e(A_a, B_a) \leq \frac{d}{n}|A_a||B_a| + \lambda\sqrt{|A_a||B_a|} \leq \frac{d}{n}|A_a||B_a| + \frac{1}{2}\lambda(|A_a| + |B_a|)$

using the arithmetic-geometric inequality in the last bullet. From here, we get

$$\begin{aligned}\frac{d}{n}\,\mathbb{E}[|A_a|^2] &\leq \mathbb{E}[e(A_a, A_a)] + \lambda|A_a|] \\ &= \rho_{aa}e(A, A) + \lambda\mu_a|A|.\end{aligned}$$

Using Proposition 20 again, we have $e(A, A) \leq \frac{d}{n}|A|^2 + \lambda|A|$. Therefore,

$$\mathbb{E}[|A_a|^2] \leq \rho_{aa}|A|^2 + \frac{n\lambda}{d}(\rho_{aa} + \mu_a)|A|. \tag{3}$$

A similar bound holds for $\mathbb{E}[|B_a|^2]$. Next, we estimate

$$\begin{aligned}\frac{d}{n}\,\mathbb{E}[|A_a||B_a|] &\geq \mathbb{E}[e(A_a, B_a) - \frac{1}{2}\lambda(|A_a| + |B_a|)] \\ &= \rho_{aa}e(A, B) - \frac{1}{2}\lambda\mu_a(|A| + |B|) \\ &\geq \rho_{aa}(\frac{d}{n}|A||B| - \lambda\sqrt{|A||B|}) - \frac{1}{2}\lambda\mu_a(|A| + |B|) \\ &\geq \rho_{aa}\frac{d}{n}|A||B| - \frac{1}{2}\lambda(\rho_{aa} + \mu_a)(|A| + |B|).\end{aligned}$$

From here,

$$\begin{aligned}\mathbb{E}[(|A_a| - |B_a|)^2] &= \mathbb{E}[|A_a|^2 - 2|A_a||B_a| + |B_a|^2] \\ &\leq \rho_{aa}|A|^2 - 2\rho_{aa}|A||B| + \rho_{aa}|B|^2 + \frac{2n\lambda}{d}(\rho_{aa} + \mu_a)(|A| + |B|).\end{aligned}$$

Since $|A| = |B|$, this simplifies to

$$\mathbb{E}[(|A_a| - |B_a|)^2] \leq \frac{4\lambda n}{d}(\rho_{aa} + \mu_a)|A|. \tag{4}$$

An analogous bound holds for $\mathbb{E}[(|A_b| - |B_b|)^2]$. Combining equations (2), (3) and (4), we conclude that

$$\begin{aligned}\Big|\,\mathbb{E}[|A_a||B_b| - |A_b||B_a|]\Big| &\leq \sqrt{\mathbb{E}[|A_a|^2]}\sqrt{\mathbb{E}[(|B_b| - |A_b|)^2]} + \sqrt{\mathbb{E}[|A_b|^2]}\sqrt{\mathbb{E}[(|A_a| - |B_a|)^2]} \\ &\leq \sqrt{\rho_{aa}|A|^2 + \frac{\lambda n}{d}(\rho_{aa} + \mu_a)|A|}\sqrt{\frac{4\lambda n}{d}(\rho_{bb} + \mu_b)|A|} \\ &\quad + \sqrt{\rho_{bb}|A|^2 + \frac{\lambda n}{d}(\rho_{bb} + \mu_b)|A|}\sqrt{\frac{4\lambda n}{d}(\rho_{aa} + \mu_a)|A|}.\end{aligned}$$

We assumed $|A| \geq \frac{\lambda n}{\xi^2 d}$, and also we have $\rho_{aa} \leq \mu_a$, $\rho_{bb} \leq \mu_b$, $\xi \leq \frac{1}{2}$, so we can simplify this bound:

$$\begin{aligned}\Big|\,\mathbb{E}[|A_a||B_b| - |A_b||B_a|]\Big| &\leq 4\sqrt{\mu_a\mu_b}\sqrt{|A|^2 + 2\xi^2|A|^2}\sqrt{2\xi^2|A|^2} \\ &\leq 4\xi\sqrt{\mu_a\mu_b}|A|^2.\end{aligned}$$

◀

Now we employ the properties of the regular pair (A, B).

▶ **Lemma 32.** *Let $\epsilon \in (0, \frac{1}{2})$, let ρ be a $\vec{G}$-realizable distribution for an orientation of an (n, d, λ)-graph G (non-bipartite, without isolated vertices) and let $A, B \subseteq V(G)$ be disjoint sets such that $\beta = \frac{\vec{e}(A,B)}{|A||B|} \geq \frac{d}{4n}$, $|A| = |B| \geq \frac{\lambda n}{\epsilon^2 d}$, and*

$$\forall A' \subseteq A, B' \subseteq B; \left|\vec{e}(A', B') - \beta|A'||B'|\right| < 8\epsilon\beta|A|^2.$$

Then

$$|\rho_{ab} - \rho_{ba}| < 20\epsilon.$$

Proof. Using the notation as above, let us define $S_a = \{v \in S : X_v = a\}$ and let us estimate $\vec{e}(A_a, B_b)$, the number of directed edges from A to B labeled (a, b). On the one hand, each directed edge gets labeled (a, b) with probability ρ_{ab}, so $\mathbb{E}[\vec{e}(A_a, B_b)] = \rho_{ab}\vec{e}(A, B)$. Similarly, $\mathbb{E}[\vec{e}(A_b, B_a)] = \rho_{ba}\vec{e}(A, B)$. On the other hand, by assumption we have

$$\left|\vec{e}(A_a, B_b) - \beta|A_a||B_b|\right| < 8\epsilon\beta|A|^2 \qquad \text{and} \qquad \left|\vec{e}(A_b, B_a) - \beta|A_b||B_a|\right| < 8\epsilon\beta|A|^2$$

for every particular choice of $A_a \subseteq A, A_b \subseteq A, B_a \subseteq B, B_b \subseteq B$. From here,

$$|\rho_{ab} - \rho_{ba}|\vec{e}(A, B) = \left|\mathbb{E}[\vec{e}(A_a, B_b) - \vec{e}(A_b, B_a)]\right| \leq \beta\left|\mathbb{E}[|A_a||B_b| - |A_b||B_a|]\right| + 16\epsilon\beta|A|^2$$

Since $|A| = |B| \geq \frac{\lambda n}{\epsilon^2 d}$, we can apply Lemma 31 with $\xi = \epsilon$, and we get

$$\left|\mathbb{E}[\big||A_a||B_b| - |A_b||B_a|\big]\right| \leq 4\epsilon\sqrt{\mu_a\mu_b}|A|^2.$$

Combining these two bounds, we get

$$|\rho_{ab} - \rho_{ba}|\vec{e}(A, B) \leq 4\beta\epsilon\sqrt{\mu_a\mu_b}|A|^2 + 16\beta\epsilon|A|^2 \leq 20\beta\epsilon|A|^2.$$

We also have $\vec{e}(A, B) = \beta|A||B| = \beta|A|^2$, so we conclude that $|\rho_{ab} - \rho_{ba}| \leq 20\epsilon$. ◀

Now we can finish the proof of Theorem 25.

Proof of Theorem 25. We assume that ρ is $\vec{G}_n$-realizable for arbitrarily large n, where $\vec{G}_n$ is an orientation of an $(n, d(n), \lambda(n))$-graph and

$$\lim_{n\to\infty} \frac{\lambda(n)}{d(n)} = 0.$$

Clearly, such graphs cannot have isolated vertices and cannot be bipartite. We prove that $|\rho_{ab} - \rho_{ba}| \leq 20\epsilon$ for every $\epsilon > 0$, which implies that ρ is symmetric.

For a given $\epsilon > 0$, let $K \geq 1$, $\gamma > 0$ be the constants given by Lemma 30. Then we choose n large enough so that $\frac{\lambda(n)}{d(n)} < \min\{\gamma, \frac{\epsilon^2}{2K}\}$, and ρ is $\vec{G}_n$-realizable on some orientation of an $(n, d(n), \lambda(n))$-graph. Lemma 30 gives a pair of disjoint sets $A, B \subseteq V$ such that $\beta = \frac{\vec{e}(A,B)}{|A||B|} \geq \frac{d}{4n}$ and

$$\forall A' \subseteq A, B' \subseteq B; \left|\vec{e}(A', B') - \beta|A'||B'|\right| < 8\epsilon\beta|A|^2.$$

Moreover, the parameters are chosen so that $|A| = |B| \geq \frac{n}{2K} \geq \frac{\lambda(n)n}{\epsilon^2 d(n)}$. Therefore, Lemma 32 applies and we conclude that $|\rho_{ab} - \rho_{ba}| \leq 20\epsilon$.

Since this holds for every $\epsilon > 0$, ρ is in fact a symmetric distribution ($\rho_{ab} = \rho_{ba}$) and we obtain that ρ is G_n-realizable for each G_n as an undirected graph. Therefore, Theorem 22 implies that ρ is an i.i.d. mix. ◀

References

1. Boaz Barak, Fernando G. S. L. Brandão, Aram W. Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In Howard J. Karloff and Toniann Pitassi, editors, *ACM STOC*, pages 307–326. ACM, 2012.
2. Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In Rafail Ostrovsky, editor, *FOCS*, pages 472–481. IEEE, 2011.
3. Bruno de Finetti. *Funzione Caratteristica di un Fenomeno Aleatorio*, pages 251–299. 6. Memorie. Academia Nazionale del Linceo, 1931.
4. Bruno de Finetti. La prévision: ses lois logiques, ses sources subjectives. *Annales de l'Institute Henry Poincaré*, 7(1):1–68, 1937.
5. Persi Diaconis. Finite forms of de Finetti's theorem on exchangeability. *Synthese*, 36(2):271–281, 1977.
6. Persi Diaconis and David Freedman. Finite exchangeable sequences. *The Annals of Probability*, pages 745–764, 1980.
7. Jair Donadelli and Yoshiharu Kohayakawa. A density result for random sparse oriented graphs and its relation to a conjecture of Woodall. *Electr. J. Comb.*, 9(1), 2002.
8. Olav Kallenberg. *Probabilistic symmetries and invariance principles*, volume 9. Springer, 2005.
9. Tali Kaufman and Alexander Lubotzky. Edge transitive Ramanujan graphs and symmetric LDPC good codes. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, pages 359–366, 2012.
10. Tali Kaufman and Alexander Lubotzky, 2014. personal communication.
11. Tali Kaufman and Avi Wigderson. Symmetric LDPC codes and local testing. In Andrew Chi-Chih Yao, editor, *ICS*, pages 406–421. Tsinghua University Press, 2010.
12. John F.C. Kingman. Uses of exchangeability. *The Annals of Probability*, pages 183–197, 1978.
13. Michael Krivelevich and Benny Sudakov. Pseudo-random graphs. *Bolyai Society Mathematical Studies*, 15:199–262, 2006.
14. Alexander Lubotzky and Roy Meshulam. A Moore bound for simplicial complexes. *Bull. London Math. Soc.*, 39:353–358, 2007.
15. John E. Maxfield and Henryk Minc. On the matrix equation X'X = A. *Proceedings of the Edinburgh Mathematical Society (Series 2)*, 13:125–129, 12 1962.
16. T.S. Motzkin and E.G. Straus. Maxima for graphs and a new proof of a theorem of Turán. *Canada J. Math.*, 17:533–540, 1965.
17. Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*. Algorithms and Combinatorics 28. Springer, 2012.
18. Ankit Sharma and Jan Vondrák. Multiway cut, pairwise realizable distributions, and descending thresholds. In David B. Shmoys, editor, *STOC*, pages 724–733. ACM, 2014.
19. A. J. Stam. Distance between sampling with and without replacement. *Statistica Neerlandica*, 32(2):81–91, 1978.
20. Luca Trevisan. Max cut and the smallest eigenvalue. *SIAM J. Comput.*, 41(6):1769–1786, 2012.
21. William T. Trotter and Peter Winkler. Ramsey theory and sequences of random variables. *Combinatorics, Probability & Computing*, 7(2):221–238, 1998.

Global and Local Information in Clustering Labeled Block Models*

Varun Kanade, Elchanan Mossel, and Tselil Schramm

University of California, Berkeley
vkanade@eecs.berkeley.edu, mossel@stat.berkeley.edu, tschramm@cs.berkeley.edu

Abstract

The stochastic block model is a classical cluster-exhibiting random graph model that has been widely studied in statistics, physics and computer science. In its simplest form, the model is a random graph with two equal-sized clusters, with intra-cluster edge probability p, and inter-cluster edge probability q. We focus on the sparse case, *i. e.*, $p, q = O(1/n)$, which is practically more relevant and also mathematically more challenging. A conjecture of Decelle, Krzakala, Moore and Zdeborová, based on ideas from statistical physics, predicted a specific threshold for clustering. The negative direction of the conjecture was proved by Mossel, Neeman and Sly (2012), and more recently the positive direction was proven independently by Massoulié and Mossel, Neeman, and Sly.

In many real network clustering problems, nodes contain information as well. We study the interplay between node and network information in clustering by studying a *labeled* block model, where in addition to the edge information, the true cluster labels of a small fraction of the nodes are revealed. In the case of two clusters, we show that below the threshold, a small amount of node information does not affect recovery. On the other hand, we show that for any small amount of information efficient local clustering is achievable as long as the number of clusters is sufficiently large (as a function of the amount of revealed information).

1998 ACM Subject Classification G.3 Probability and Statistics

Keywords and phrases stochastic block models, information flow on trees

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.779

1 Introduction

The stochastic block model is one of the most popular models for networks with clusters. The model has been extensively studied in statistics [14, 27, 4], computer science (where it is called the planted partition problem) [10, 15, 7, 19] and theoretical statistical physics [8, 29, 9].

The simplest block model has k clusters of equal size, and is generated as follows. Starting with n nodes, each node v is randomly assigned a label σ_v from the set $\{1, \ldots, k\}$. For each pair of nodes, (u, v), if their labels are identical an edge is added between them with probability p, otherwise an edge is added with probability q. Often the case when $p > q$ is considered, and the question of interest is understanding how large $p - q$ must be for correct clusters recovery to be possible. In the recovery problem the input consists of the unlabeled graph and the desired output is a partition of the graph.

* Varun Kanade is supported by a Simons Postdoctoral Fellowship, Elchanan Mossel acknowledges the support of the NSF (grants DMS 1106999 and CCF 1320105) and ONR (DOD ONR grant N000141110140), and Tselil Schramm is supported by a Berkeley Chancellor's Fellowship and the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 1106400.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 779–792

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

Real world networks are typically sparse. Thus, an interesting setting in the block model is when p and q are in $O(1/n)$. Here, it is more convenient to parametrize the problem by setting $p = a/n$ and $q = b/n$, where a, b are constants. In the sparse setting, exact recovery is impossible as the resulting graph will have isolated nodes. Moreover, it is easy to see that even nodes with constant degree cannot be classified accurately given all other nodes in the graph. Thus the goal is to find a partition that has non-trivial correlation with the original clusters (up to permutation of cluster labels). This has sometimes been referred to as the *cluster detection* problem (see e.g. [8]); throughout the paper we refer to it as the *cluster recovery* problem (though note that the goal is not to recover every cluster with probability 1).

General results of Coja-Oghlan [6] imply that it is possible to identify a partition that is correlated with the true hidden partition when $(a-b)^2 \geq Ck^4(a+(k-1)b)$. A beautiful physics paper by Decelle *et al.* [8] conjectured that the recovery problem is feasible for the case of two clusters when $(a-b)^2 > 2(a+b)$ and impossible when $(a-b)^2 < 2(a+b)$. The non-reconstructability in the case where $(a-b)^2 < 2(a+b)$ was proved by Mossel, Neeman and Sly [22], and more recently the same authors [24] and Massoulié [18] independently showed that recovery is possible when $(a-b)^2 > 2(a+b)$.

1.1 The Labeled Stochastic Block Model

The aforementioned results along with previous results for denser block models provide a detailed picture of recovery in the stochastic block model. However, the model they consider is idealized and does not capture many aspects of real network problems. One such aspect is that in many realistic settings, node label information is available for some of the nodes. For example, in social networks, the group label of some individuals (nodes) is known. In metabolic networks, the function of some of the nodes may be known. Indeed, there has been much recent work in the machine learning and applied networks communities on combining node and network information (see for example [5, 2, 3]). There are several ways in which node and edge information can be incorporated; in real applications nodes and edges contain rich information which is noisy, but correlated with the node's "true" label and with the "similarity" of pairs of nodes.

In this paper, we study a simple model which incorporates both node and edge information which we call the *labeled* stochastic block model. This model has been considered previously in the physics literature [8, 28, 1]. In addition to having the unlabeled graph as an input, a *small* random fraction of the nodes' labels are also provided as input to the clustering algorithm.

1.2 The Big Effect of a Small Number of Node Labels

It is easy to see that even a vanishing fraction of node labels can play a major role in the cluster recovery problem. For example, consider the denser case where the clusters $C_1, \ldots, C_k$ can be identified accurately [19]. Here, it is impossible to distinguish between a clustering $C_1, \ldots, C_k$ where the nodes in cluster C_i have label i and the same clustering where the nodes in cluster i have label $\pi(i)$ for any permutation π of the labels. However, note that for any $p > 0$, given a p-fraction of the node labels, it is possible to identify the permutation π correctly with high probability. It is natural to ask if the same result holds in the sparse case, and it is not hard to see that a similar statement can be made (see Proposition 14).

The above observation shows that even a small amount of node information can overcome the problems of symmetry in the stochastic block model. Another problem of symmetry

present in the unlabeled model is that there is no *local algorithm* that can identify clusters better than random guessing. Informally, a local algorithm determines the label of a node based solely on an $o(\log n)$ neighborhood of that node, including possibly uniform independent random variables attached to each node of the graph (see A.2 for a formal definition and [17, 13] for examples). The proof that a local algorithm cannot detect better than random guessing in this case is folklore, and we include it (in the full version [16]) for completeness. This limitation in detection may be compared to the problem of finding independent sets, where local algorithms can have non-trivial power (while still being less powerful than global algorithms) [12]. It is therefore natural to ask:

▶ Question 1. Does a vanishing fraction of labeled nodes allow *local algorithms* to detect clusters? If so, when?

An even a more direct question relates to the statistical power of revealing some of the node labels. While it is clear that revealing a large fraction of the node labels allows non-trivial recovery, it is far from clear what the effect is when this fraction is vanishingly small. On the one hand, we might expect by continuity that revealing a vanishing fraction of the node labels will be identical in the limit to revealing no labels. On the other hand, we might imagine how a small fraction of the node labels could be used as seeds for recovery algorithms. We thus ask:

▶ Question 2. Does revealing a vanishing fraction of the node labels change the detectability threshold? Does it change the fraction of correctly labeled nodes?

The latter question was considered in recent work in statistical physics [30, 28, 1].

1.3 Our Results

To set the stage for our contributions, we begin with some observations regarding the utility of local information. More formal versions of these propositions are provided in Appendix A. The proofs of these propositions are straightforward (provided in the full version [16]), but they are useful for establishing context of how information about (a small fraction of) node labels may help. The first is that even a vanishingly small proportion of node labels aids in breaking the symmetry and assigning labels to the cluster assignments.

▶ **Proposition 1** (Informal version). *Given a clustering algorithm which outputs clusters correlated with the true clustering, a small fraction of revealed node labels is sufficient to output a labeling which is correlated with the true labeling.*

In the absence of any node information, it is an easy folklore result that any local algorithm cannot recover clusters. However, we show that in the case of two clusters, when a small fraction of node labels are revealed, a local algorithm is able to recover the clusters optimally. This latter result is a direct corollary of a robust reconstruction result on trees of [23].

▶ **Proposition 2** (Informal version). *In the unlabeled stochastic block model, no local algorithm can find a clustering correlated with the true clustering.*

▶ **Proposition 3** (Informal version). *In an instance of the labeled stochastic block model, when $k = 2$, if $(a-b)^2 > C(a+b)$ for some large constant C, then there is a local algorithm which given a vanishing fraction of labeled nodes, reconstruct the label of all nodes with the same accuracy as the optimal (non-local) algorithm for the unlabeled problem.*

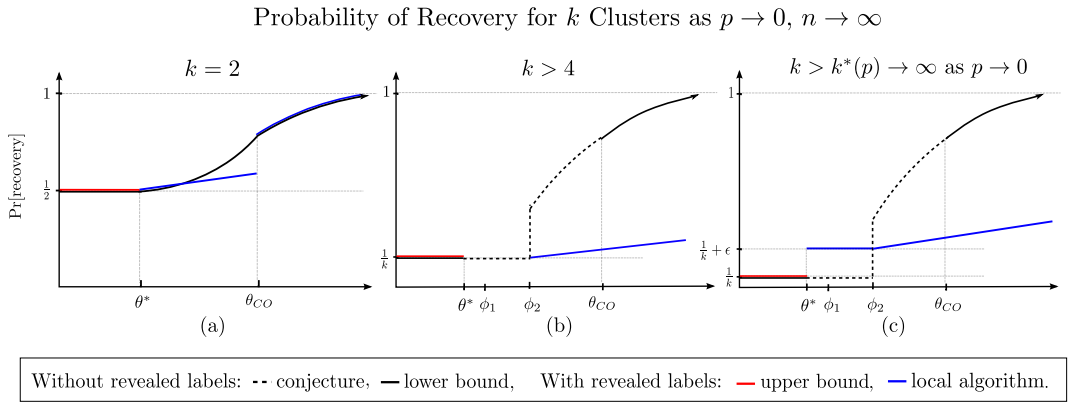


Figure 1 Previous work (black) and our contributions (colored). The x-axes represent the second eigenvalue of the corresponding broadcast process on the coupled Galton-Watson Tree when the average degree is fixed–in simpler terms, this is an increasing function of the ratio $\frac{a-b}{a}$. In all three cases, θ^* is the reconstruction threshold corresponding to the root reconstruction problem on trees, and θ_{CO} is the threshold of [6]. In the two-cluster case (Subfigure (a)), θ^* corresponds exactly to the Kesten-Stigum bound of $(a-b)^2 < 2(a+b)$ [8, 22]. For the case of larger k, $\theta^* < (a-b)^2/k(a+(k-1)b)$ (see Subfigures (b), (c) and Proposition 10). We prove analogously that recovery is not possible below θ^* in the labeled model as $p \to 0$ for all k (Theorem 11). In the two-cluster case, recent results of [24] and [18] show that recovery is possible in the range (θ^*, θ_{CO}); above θ_{CO}, a combination of the results of [6] and [23] give optimal recovery in the standard model for $k = 2$; we observe that in the labeled model for $k = 2$, one can reconstruct better than randomly in the range (θ^*, θ_{CO}) and optimally above θ_{CO} using *local* algorithms (see Propositions 19 and 18). The results of [6] also give non-trivial recovery guarantees above θ_{CO} for all k. In the k-cluster case (Figures (b), (c)), the picture is more complicated: ϕ_1 and ϕ_2 are conjectured brute-force and efficient solvability thresholds respectively, both conjectured by [8]—above ϕ_1 recovery is possible via brute-force enumeration, and above ϕ_2 an efficient algorithm for recovery exists. Above ϕ_2, Proposition 19 shows that recovery is possible for k clusters via a *local* algorithm. In Subfigure (c), for any b, p, if $k > k^*(p)$ and $(a-b)/k > 1$, as in Theorem 8, we give an efficient *local* recovery algorithm that correctly labels $\frac{1}{k} + \epsilon$ of the nodes, even below the conjectured efficient recovery threshold ϕ_2.

We also observe that results on census reconstruction [25] imply that above the Kesten-Stigum bound a vanishingly small fraction of revealed nodes suffices for the cluster recovery problem.

▶ **Proposition 4** (Informal version). *For any fixed k, above the robust reconstruction threshold (i.e. when $(a-b)^2 > k(a+(k-1)b)$), when the fraction of revealed node labels is vanishingly small, the cluster recovery problem is solvable.*

In this context, one might expect that labels could allow clustering in the labeled model in regimes which cannot be effectively clustered in the unlabeled model. The case of two clusters is the case we understand the best. Here, utilizing results for the reconstruction problem on trees and of [22], we answer Question 2 in the *negative* (Theorem 5) and at the same time answer Question 1 positively (Propositions 18 and 19). The complete picture for the case of two clusters is presented in Figure 1(a).

For any fixed $k > 2$, the picture is much more complicated. In this case, we observe that below the tree reconstruction threshold (this corresponds to θ^* in Figure 1(b)), a vanishing fraction of node labels do not assist in the cluster recovery problem (see Theorem 5 in Section 4).

▶ **Theorem 5** (Informal version). *For any fixed k, below the associated tree reconstruction threshold (to be defined later), when the fraction of revealed node labels is vanishingly small, the cluster recovery problem is not solvable. In particular, when $k = 2$, the threshold is the Kesten-Stigum bound of $(a-b)^2 < 2(a+b)$; for $k \geq 2$, if $a - b < k$ then recovery is impossible.*

Our main interest is in the case when the number of clusters is very large. Here, we consider the setting when the fraction of revealed nodes $p \to 0$, and simultaneously the number of clusters $k = k(p) \to \infty$. In this setting, we show that revealing node labels has a dramatic effect on the threshold for cluster recovery. We show that a local algorithm successfully solves the cluster recovery problem even below the conjectured algorithmic threshold in the unlabeled case, $(a-b)^2 = k(a+(k-1)b)$. As the number of clusters $k \to \infty$, our algorithm works all the way down to the tree reconstruction threshold of $(a-b)/k > 1$. Moreover, it is impossible to recover (locally or globally) with a vanishing fraction of labeled nodes if $(a-b)/k < 1$. Both results follow from the corresponding results on trees.

▶ **Theorem 6** (Informal version). *For every $p > 0$, if k is large enough as a function of p, and $a - b > (1+\delta)k$, then the label of a random node can be recovered with probability at least $\frac{1}{k} + \epsilon$, where ϵ depends on δ but is independent of p.*

We give a more formal statement of Theorem 8 in Section 3.

Recent work in statistical physics [30] argues that for every *fixed number of clusters* k, a vanishing fraction of labels does not provide any advantage in the detection probability over having no labels at all. We note that in our results, the order of limits is exchanged as the number of clusters k needed for our results to hold, depends on the fraction of nodes revealed. Thus, there is no contradiction between the results (see also [1, 28]). Figure 1(c) provides a detailed picture of the case in which the number of clusters is very large (in the setting of Theorem 6).

Open Problems

In the case of two clusters, we conjecture that whenever any fraction of node labels are revealed, there is a *local algorithm* that recovers the clusters optimally. This would follow from a related conjecture regarding information flow on trees stated below. We report some simulations suggesting the veracity of the conjecture in Appendix B.

▶ **Conjecture 7** (Informal version). *Let T be an infinite tree with root ρ. The tree is labeled from the set $\{\pm 1\}$ as follows. First, the root is assigned a label from $\{\pm 1\}$ at random. Along each edge the label is propagated with probability $1-\eta$ and flipped with probability η. Let (T, τ) denote the resulting labeled tree. Add each node independently to a set R with probability p. Finally for any r, let ∂T_r denote the set of leaves at depth r. Then, for any value of $p > 0$ and $\eta < 1/2$,*

$$\lim_{r\to\infty} \mathbb{E}\left| \Pr[\tau_\rho = 1 \mid \tau_R] - \Pr[\tau_\rho = 1 \mid \tau_R, \tau_{\partial T_r}] \right| = 0$$

In addition to Conjecture 7, several interesting questions remain, particularly in the regime where k is large. When k is large, is it possible to use global and local information together to obtain better recovery guarantees? Which algorithmic tools might allow one to use global and local information simultaneously?

Another open problem relates to different types noise models. The assumption in the current paper is that each label is revealed accurately with a vanishing probability. But one may consider other types of noise. In particular, we may assume for example that for each

node independently we are given the correct label with small probability δ and otherwise a uniformly chosen label. Is it true that the same results hold for this noise model as for the noise model considered here? For most of the results presented here, it is easy to see that the answer is yes. However, for one of our main results, Theorem 6, the proof *does not* extend to the latter noise model. It is an interesting open problem to determine the effect of the noisy information in this setup.

2 Model

2.1 Stochastic Block Model

The stochastic block model is a generative model for modular random networks, defined by the following set of parameters: the number of clusters k, the expected fraction of nodes in each cluster i, $\langle f_i \rangle_{i=1}^{k}$, and a $k \times k$ symmetric affinity matrix $P_{i,j}$ indicating the edge probability between nodes of type i and j. A random network G on n nodes is generated as follows:

1. First, each node v is assigned a label $\sigma_v \in \{1, \ldots, k\}$, s.t. $\Pr[\sigma_v = i] = f_i$.
2. For every pair of nodes u, v, an edge is added between them with probability P_{σ_u, σ_v}, independently for each pair.

In this work, we are mainly interested in the sparse case, *i. e.*, when the average degree of the graph is constant. We focus on the setting where edge probabilities only depend on whether the labels of the endpoint are same or different. Thus, $P_{ii} = a/n$ for $1 \leq i \leq k$ and $P_{ij} = b/n$ for $i \neq j$, for constants $a > b$.[1] Also, we focus on the case where $f_i = 1/k$ for each i, *i. e.*, each cluster is roughly of the same size. The model is denoted by $\mathcal{G}(n, k, a, b)$, and $(G, \sigma) \sim \mathcal{G}(n, k, a, b)$ denotes an instance of a graph generated according to the model, where σ are the cluster labels of the nodes.

Labeled Block Model: The labeled block model has an additional parameter p, which is the probability with which the true cluster label of any given node is revealed. Thus, if $(G, \sigma) \sim \mathcal{G}(n, k, a, b)$ is an instance of the block model, $R \subseteq [n]$ is chosen by placing each node of G in R independently with probability p. We denote this by $(G, \sigma, R) \sim \mathcal{G}(n, k, a, b, p)$. The clustering algorithm has access to the edges of G and the cluster labels σ_R of nodes in R, *i. e.*, (G, R, σ_R).

We also introduce the following notation for convenience. For any two nodes $u, v \in G$, let $d(u, v)$ denote the distance between u and v. We let $G_r(v) = \{u \in G \mid d(u, v) \leq r\}$ denote the neighborhood of radius r around v; at times we will use G_r when v is clear from context. Let $\partial G_r(v) = \{u \in G \mid d(u, v) = r\}$ denote the boundary of $G_r(v)$.

Cluster Recovery: The *cluster recovery* problem is the problem of recovering the cluster label of nodes in the stochastic block model or labeled stochastic block model with better-than-random probability. Note that correct recovery of all nodes is not the aim, nor is it possible due to the sparsity of the graph. This problem has also been called the *cluster detection* problem and the *cluster reconstruction* problem; for consistency we will use the term recovery throughout the paper when referring to graphs, and use reconstruction when referring to broadcast processes on trees.

[1] This is the so-called assortative model.

▶ **Algorithm 1.**
Input: $(G, R) \sim \mathcal{G}(n, k, a, b, p)$, *radius* r, *max-degree* D, *revealed cluster labels* σ_R

For each node $v \notin R$
1. *Let* $G_r(v)$ *denote the (tree-like) neighborhood of* v *up to distance* r
2. *From* $G_r(v)$ *delete every subtree rooted at a node with degree larger than* D
3. *Let* L *denote the set of labels* $l \in \Sigma$ *for which there exist* $x, y \in R$ *such that* $\sigma_x = \sigma_y = l$, $d(x, v) = d(y, v) = r$, *and* v *is* x *and* y*'s first common ancestor*
4. *Assign a random label from* L *to node* v

2.2 Information Flow on Trees

We use some results regarding information flow on trees. For a detailed survey on this topic, the reader is referred to [21].

Let T be an infinite rooted tree, with the root note denoted by ρ. A Galton-Watson tree is obtained by starting with a root node, ρ, and recursively adding offspring drawn from some distribution D with mean d. In particular, we will often be interested in the case when D is Poisson(d). For any node $v \in T$, let $d(v, \rho)$ denote the distance of v from the root. Throughout the paper, we denote $T_r = \{v \in T \mid d(v, \rho) \leq r\}$ as the subtree of T up to depth r, and $\partial T_r = \{v \in T \mid d(v, \rho) = r\}$ as the boundary at depth r.

Broadcast Process: Let T be an infinite rooted tree with root ρ. Each node in the tree is assigned a label from some finite alphabet $\Sigma = \{1, \ldots, k\}$. The root is labeled by choosing a label $\tau_\rho \in \Sigma$ uniformly at random. For any edge (u, v), with $d(u, \rho) < d(v, \rho)$, τ_v is conditionally independent given τ_u, and is chosen as follows: $\tau_v = \tau_u$ with probability $1-(k-1)\eta$, and $\tau_v \in \Sigma \setminus \{\tau_u\}$ randomly otherwise, where $\eta < 1/k$ is the broadcast parameter. We denote this process by $\mathcal{T}(T, k, \eta)$ and an instance generated according to this process by $(T, \tau) \sim \mathcal{T}(T, k, \eta)$. As in the block model, we can consider the process when the label of each node is revealed with probability p, *i. e.*, $R \subseteq T$ is obtained by adding each $v \in T$ to R independently with probability p. We denote this process by $(T, \tau, R) \sim \mathcal{T}(T, k, \eta, p)$. The *reconstruction problem* is to identify the label of the root, ρ given the labeled nodes up to some depth r. Thus, the algorithm has access to (T_r, R_r, τ_{R_r}), where R_r denotes $T_r \cap R$.

Percolation Process: Let T be an infinite rooted tree with root ρ. For percolation parameter λ, each edge $e \in T$ is deleted independently with probability λ. Let $C(\rho)$ denote the component of T containing the root after percolation.

3 Recovery in the Many Clusters Regime

We show that when the number of clusters is very large, even a very small fraction of revealed node labels allow for cluster recovery, and even in some regimes below the conjectured algorithmic threshold in the standard model. More formally, if p is the probability that the label of a node is revealed, and if the number of clusters is at least $k^* = k(p)$, then even as $p \to 0$, the algorithm performs better than random assignment. The algorithm (Algorithm 1) is simple and *local*—it considers a neighborhood around each node and uses the revealed node information in the neighborhood to make its prediction.

▶ **Theorem 8.** *Let $b > 1$ be fixed, let $a = b + (1+\delta)k$ for some $\delta > 0$, let $p > 0$ be fixed. Then, there exists an $\epsilon = \epsilon(b,\delta)$ and $k^* = k^*(b,\delta,p)$, such that for every $k \geq k^*$, if $(G, R, \sigma_R) \sim \mathcal{G}(n,k,a,b,p)$, Algorithm 1 labels any random node of G correctly with probability at least ϵ. In particular, there exists settings where $(a-b)^2 < k(a+(k-1)b)$ and recovery is still possible.*

We give a proof of Theorem 8 in the full version [16]; here we give a high-level idea of the proof. First, we utilize a coupling between local neighborhoods in $\mathcal{G}(n,k,a,b)$ and a broadcast process on a rooted Galton-Watson tree with offspring distribution Poisson($\frac{a+(k-1)b}{k}$). Fix $v \in [n]$ and let $(G,\sigma) \sim \mathcal{G}(n,k,a,b)$. For large values of n, and when r is not too large (though increasing as a function of n), $G_r(v)$ looks like a tree. The degree distribution of any node in G is Binomial($n, \frac{a+(k-1)b}{kn}$) $\approx$ Poisson($\frac{a+(k-1)b}{k}$). If $\eta = \frac{b}{a+(k-1)b}$, the distribution (G_r, σ_{G_r}) resembles the distribution (T_r, τ_r), where $(T,\tau) \sim \mathcal{T}(T,k,\eta)$ corresponds to the broadcast process on a Galton-Watson tree process T with offspring distribution Poisson($\frac{a+(k-1)b}{k}$). This coupling was formally proved in [22].

▶ **Lemma 9** ([22]). *Let $r < r(n) = \frac{1}{10\log(2(a+(k-1)b))}\log(n)$. There exists a coupling between (G,σ) and (T,τ) such that $(G_r,\sigma_{G_r}) = (T_r,\tau_{T_r})$ a.a.s.*

In [20] it is shown that for larger alphabet sizes, $d(1-k\eta)^2 \geq 1$ is not the threshold for reconstruction for regular trees. As our results show, this is also the case for Galton-Watson trees. In order to understand the intuition behind Algorithm 1, it is useful to consider an *infinite color* broadcast process on a tree. Let $\tilde{\eta} \ll 1$ be a small broadcast parameter. Suppose the root ρ is given some color, which is propagated away from the root as follows. With $(1-\tilde{\eta})$ probability the neighboring node gets the same color, with $\tilde{\eta}$ probability the neighboring node gets a completely new color. The color of each node is revealed with probability p. Consider the following event: there are two nodes in the tree with the same color, for which the root ρ is the first common ancestor. If such an event occurs, this color *must* also be the color of the root. We show that this infinite-color picture is more or less accurate when k is large enough.

4 Upper Bounds Below the Threshold

In this section, we consider the setting where there are a fixed number of clusters and the fraction of revealed node labels is vanishingly small. We show that below a certain threshold that arises from the reconstruction problem on trees, in the limit as $p \to 0$, cluster recovery is not possible. We first note that a threshold exists for the tree problem.

▶ **Proposition 10.** *Let T be a Galton-Watson tree with average degree $d > 1$. Let $(T,\tau) \sim \mathcal{T}(T,k,\eta)$ be the labels obtained by the broadcast process with parameter η. There there exists a predicate, $\pi_k(d,\eta)$, monotonically decreasing in η and monotonically increasing in d, such that if $\pi_k(d,\eta)$ is false, then for each $i \in [k]$,*

$$\lim_{r\to\infty} \Pr\left[\tau_\rho = i \mid \tau_{\partial T_r}\right] \to \frac{1}{k}, \quad a.a.s.$$

For the case of $k = 2$, the exact form of π_2 is known, $\pi_2(d,\eta) = \mathbb{1}[d(1-2\eta)^2 > 1]$, which follows from [11]. In [26], the exact threshold is given for $k = 3$, and bounds on the thresholds are given for $k \geq 5$. For $k \geq 4$, the exact form π_k is not known, but it holds that if $(1-k\eta)d < 1$, $\pi_k(d,\eta)$ is false. (This was proved for the case of regular trees in [20]; the proof for Galton-Watson trees is essentially identical). For all k, a reconstructability

threshold in η, d provably exists in the limit as $n \to \infty$; the proof of Proposition 10 relies on the monotonicity of π_k in η and d, and the existence of points where reconstruction is feasible and also points where it is impossible.

The threshold from Proposition 10 can be translated to an equivalent threshold $\theta_k(a, b)$ in the stochastic block model. We show that even in the labeled stochastic block model (where each node's label is revealed with probability p), if p is small and θ_k is false then it is impossible to recover node labels with better accuracy than random guessing. Specifically, we study the setting where k is fixed, θ_k is false, and $p \to 0$. We first prove this for the general k-cluster case, then give an alternative proof for the case of two clusters (which results in a more explicit dependence on p).

▶ **Theorem 11.** *Fix $v \in [n]$, and let $(G, R, \sigma) \sim \mathcal{G}(n, k, a, b, p)$, for $a + (k-1)b > k$. Then if the predicate $\theta_k(a, b) = \pi_k(\frac{a+(k-1)b}{k}, \frac{b}{a+(k-1)b})$ is not satisfied, then for all $i \in \Sigma = [k]$,*

$$\lim_{p \to 0} \lim_{n \to \infty} \Pr[\sigma_v = i | G, R, \sigma_R] = \frac{1}{k}, \quad a.a.s.$$

The above result says that as the amount of revealed node information goes to zero, recovering a clustering that is correlated with the true clustering is not possible if θ_k is false. The proof of Theorem 11 is given in the full version [16], but we give a high-level overview of the proof here.

We again utilize a coupling between local neighborhoods in $\mathcal{G}(n, k, a, b)$ and a broadcast process on a rooted Galton-Watson tree. As in Section 3, let T be a Galton-Watson tree with offspring distribution Poisson($\frac{a+(k-1)b}{k}$) and broadcast parameter $\eta = \frac{b}{a+(k-1)b}$. We fix $v \in [n]$ and let $(G, \sigma) \sim \mathcal{G}(n, k, a, b)$. The distribution $(G_r(v), \sigma_{G_r(v)})$ resembles the distribution (T_r, τ_r).

We also use a result of [22] which states that conditioned on $\sigma_{\partial G_r}$, information from further nodes is not helpful in clustering.

▶ **Lemma 12** ([22]). *Fix $v \in [n]$, and let $(G, R, \sigma) \sim \mathcal{G}(n, k, a, b, p)$, with $a + (k-1)b > k$. For $r \leq \frac{1}{10 \log(2(a+(k-1)b))} \log n$, let $C = \{u \in G \mid d(u, v) > r\}$, $B = \partial G_r$, and $A = \{u \in G \mid d(u, v) \leq r\}$. Then*

$$\Pr[\sigma_A \mid \sigma_B, \sigma_C, G] = (1 + o(1)) \Pr[\sigma_A \mid \sigma_B, G].$$

In [22], the lemmas above are stated for the case when $k = 2$; however, the same proofs apply for any value of k. Armed with Lemmas 9, 12 and Proposition 10, we can prove Theorem 11 by choosing p small enough that there is no label information in the local neighborhood of any vertex with high probability, then showing that the global graph information is not helpful in recovering the labels.

In the special case of $k = 2$ clusters, it is possible to prove the same result using a slightly different technique. Here, we get a more explicit convergence rate in terms of p. Note that the RHS in the statement of Theorem 13 cannot be smaller than p, since with probability p the node of the label itself is revealed.

▶ **Theorem 13.** *Fix $v \in [n]$, and let $(G, R, \sigma) \sim \mathcal{G}(n, 2, a, b, p)$, for $a + b > 2$. Then if $(a-b)^2 < 2(a+b)$, then*

$$\lim_{n \to \infty} \mathbb{E} \left| \Pr[\sigma_v = 1 \mid G, R, \sigma_R] - \frac{1}{2} \right| \leq \frac{1}{2} \sqrt{\frac{p}{1 - \frac{(a-b)^2}{2(a+b)}}}$$

We give a proof of this better dependence in the full version [16].

Acknowledgments. E. M. thanks Cris Moore, Joe Neeman, Allan Sly and Lenka Zdeborová for many interesting discussions related to the block model. We would like to thank the authors of [30] for discussion of their work at its early stages. The authors would like to thank the Simons Institute for the Theory of Computing where much of the work reported here was carried out.

References

1 Armen E. Allahverdyan, Greg Ver Steeg, and Aram Galstyan. Community detection with and without prior information. *Europhysics Letters*, 90:18002, 2010.

2 Sugato Basu, Arindam Banerjee, and Raymond J Mooney. Semi-supervised clustering by seeding. In *ICML*, volume 2, pages 27–34, 2002.

3 Sugato Basu, Mikhail Bilenko, and Raymond J. Mooney. A probabilistic framework for semi-supervised clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 59–68. ACM, 2004.

4 P. J. Bickel and A. Chen. A nonparametric view of network models and Newman-Girvan and other modularities. *Proceedings of the National Academy of Science*, 106(50):21068–21073, 2009.

5 Olivier Chapelle, Jason Weston, and Bernhard Schoelkopf. Cluster kernels for semi-supervised learning. In *NIPS*, pages 585–592, 2002.

6 A. Coja-Oghlan. Graph partitioning via adaptive spectral techniques. *Combinatorics, Probability and Computing*, 19(02):227–284, 2010.

7 A. Condon and Richard M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Structures and Algorithms*, 18(2):116–140, 2001.

8 Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Phys. Rev. E*, 84:066106, Dec 2011.

9 Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Inference and phase transitions in the detection of modules in sparse networks. *Phys. Rev. Lett.*, 107:065701, 2011.

10 Martin E. Dyer and Alan M. Frieze. The solution of some random NP-hard problems in polynomial expected time. *Journal of Algorithms*, 10(4):451–489, 1989.

11 William Evans, Claire Kenyon, Yuval Peres, and Leonard J. Schulman. Broadcasting on trees and the Ising model. *The Annals of Applied Probability*, 10(2):410–433, 2000.

12 David Gamarnik and Madhu Sudan. Limits of local algorithms over sparse random graphs. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 369–376. ACM, 2014.

13 Hamed Hatami, László Lovász, and Balázs Szegedy. Limits of local-global convergent graph sequences. *arXiv preprint arXiv:1205.4356*, 2012.

14 P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

15 Mark Jerrum and G. B. Sorkin. The Metropolis algorithm for graph bisection. *Discrete Applied Mathematics*, 82(1–3):155–175, 1998.

16 Varun Kanade, Elchanan Mossel, and Tselil Schramm. Global and local information in clustering labeled block models. Available at `http://arxiv.org/abs/1404.6325`, 2014.

17 Russell Lyons and Fedor Nazarov. Perfect matchings as iid factors on non-amenable groups. *European Journal of Combinatorics*, 32(7):1115–1125, 2011.

18 Laurent Massoulié. Community detection thresholds and the weak Ramanujan property. In *Proceedings of the Symposium on the Theory of Computation (STOC)*, 2014.

19 Frank McSherry. Spectral partitioning of random graphs. In *Proceedings of IEEE Conference on the Foundations of Computer Science (FOCS)*, pages 529–537, 2001.

20 Elchanan Mossel. Reconstruction on trees: Beating the second eigenvalue. *The Annals of Applied Probability*, 11(1):285–300, 2001.

21 Elchanan Mossel. Survey: Information flow on trees. Available at `arxiv.org/abs/math/0406446`, 2004.

22 Elchanan Mossel, Joe Neeman, and Allan Sly. Stochastic block models and reconstruction. Preprint avaiable at `arxiv.org/abs/1202.1499`, 2012.

23 Elchanan Mossel, Joe Neeman, and Allan Sly. Belief propogation, robust reconstruction, and optimal recovery of block models. Preprint available at `arxiv.org/abs/1309.1380`, 2013.

24 Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. Preprint available at `http://arxiv.org/abs/1311.4115`, 2013.

25 Elchanan Mossel and Yuval Peres. Information flow on trees. *Ann. Appl. Probab.*, 13(3):817–1230, 2003.

26 A. Sly. Reconstruction of symmetric potts models. In *Proceedings of the 41st ACM Symposium on Theory of Computing*, pages 581–590, 2009.

27 T. A. B. Snijders and K. Nowicki. Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of Classification*, 14(1):75–100, 1997.

28 Greg Ver Steeg, Cristopher Moore, Aram Galstyan, and Armen E. Allahverdyan. Phase transitions in community detection: A solvable toy model. Available at `http://www.santafe.edu/media/workingpapers/13-12-039.pdf`, 2013.

29 Pan Zhang, Florent Krzakala, Jörg Reichardt, and Lenka Zdeborová. Comparitive study for inference of hidden classes in stochastic block models. *Journal of Statistical Mechanics: Theory and Experiment*, 2012.

30 Pan Zhang, Cristopher Moore, and Lenka Zdeborová. Phase transitions in semisupervised clustering of sparse networks. Available at `http://arxiv.org/abs/1404.7789`, 2014.

A When Little Information Helps

Here, we give formal statements of the simple observations described in Section 1 which illustrate the power and limitations of revealed labels in the stochastic block model.

A.1 Revealed Labels and Cluster Labeling

▶ **Proposition 14.** *Let $C : [n] \to [k]$ be the output of some clustering algorithm with the guarantee that there exists a permutation $\pi : [k] \to [k]$ such that*

$$\frac{1}{n}\sum_i \mathbb{1}[\pi(C(i)) = \sigma_i] \geq \frac{1}{k} + \epsilon,$$

Then for $p \geq \frac{1}{n}\frac{512k}{\epsilon^3}\log\frac{4k}{\delta}$, if a p-fraction of node labels are revealed, we can find a function $g : [k] \to [k]$ such that

$$\frac{1}{n}\sum_i \mathbb{1}[g(C(i)) = \sigma_i] \geq \frac{1}{k} + \frac{\epsilon}{2}$$

with probability at least $1 - \delta$.

The proof follows easily from the following lemma, which is a simple application of the Chernoff-Hoeffding bound.

▶ **Lemma 15.** *Let D be a probability distribution over $[k]$, and let $S \sim D^m$ be a sample. When $m \geq \frac{64}{\epsilon^2}\log(\frac{4k}{\delta})$, for $i = \text{plurality}(S)$ (ties may be broken arbitrarily), with probability at least $1 - (\delta/2)$,*

$$|D_i - \max_j D_j| \leq \frac{\epsilon}{4},$$

where D_j is the probability of j under D.

The complete proofs are given in the full version [16].

A.2 Limitations of Local Algorithms in the Unlabeled Model

Now, we discuss the impact of revealed labels in the context of local algorithms. We use the definition of local algorithms as in [12]. (The reader is referred to their paper and references therein for more background on local algorithms.)

▶ **Definition 16.** Let G be a graph with node set V, and for each $v \in V$, let $X_v \in [0,1]$ uniformly at random. An *r-local algorithm* on G is one in which the value of each node $v \in V$ is decided by a function $f_v(G_r(v), X_r(v))$, where $X_r(v)$ is the set of samples from D associated with $G_r(v)$.

The proposition below formalizes the intuitive statement that no r-local algorithm can accurately reconstruct clusters in the unlabeled stochastic block model for $r = o(\log n)$. The proof is provided in the full version [16].

▶ **Proposition 17.** *In the unlabeled stochastic block model, let A be a local algorithm with node functions $\{f_v\} : G_r(v) \to \Sigma$, where here $G_r(v)$ denotes the structural information and random variables on the neighborhood of radius $r = o(\log n)$ around v. Then for all $\epsilon > 0$,*

$$\lim_{n\to\infty} \Pr_{G,X}[\max_{\pi} \frac{1}{n}\sum_{v} 1(f_v(G_r(v)) = \pi(\sigma_v)) \geq \frac{1}{k} + \epsilon] = 0,$$

where the maximum is taken over all possible permutations of the labels.

A.3 Optimal Local Reconstruction in the Labeled Model when $k = 2$

Before giving a formal statement of Proposition 18, we need to introduce some notation related to broadcast processes on trees. Let $(T, \tau) \sim \mathcal{T}(T, 2, \eta)$, where T is a Galton-Watson tree with offspring distribution Poisson(d). Let

$$\mathsf{T}^*(d,\eta) = \lim_{r\to\infty} \mathbb{E}\left|\Pr[\tau_\rho = 1 \mid \tau_{\partial T_r}] - \frac{1}{2}\right|$$

It follows from the work of Evans *et al.* that $\mathsf{T}^*(d,\eta) > 0$ if and only if $d(1-2\eta)^2 > 1$ [11].

Mossel *et al.* [23] looked at the robust reconstruction problem on trees. Let $(T,\tau) \sim \mathcal{T}(T,2,\eta)$ be as defined above. For some parameter $\delta \in [0, 1/2)$, let $\tilde{\tau}_u$ be the random variable, such that $\tilde{\tau}_u = \tau_u$ with probability $1-\delta$, and $\tilde{\tau}_u = 1 - \tau_u$ with probability δ. In [23], the authors consider the question of reconstruction of the root label given the noisy labels, $\tilde{\tau}_{\partial T_r}$, in the limit as $r \to \infty$. They showed that if

$$\widetilde{\mathsf{T}}^*(d,\eta) = \lim_{r\to\infty} \mathbb{E}\left|\Pr[\tau_\rho = 1 \mid \tilde{\tau}_{\partial T_r}] - \frac{1}{2}\right|,$$

then for any $\delta \in [0, 1/2)$, whenever $d(1-2\eta)^2 \geq C$ for a sufficiently large constant C, $\widetilde{\mathsf{T}}^*(d,\eta) = \mathsf{T}^*(d,\eta)$.

▶ **Proposition 18.** *Let $(G, R, \sigma_R) \sim \mathcal{G}(n, 2, a, b, p)$, with $a + b > 2$. Then, there exists a large constant C, such that if $(a-b)^2 > C(a+b)$, there is a local algorithm A such that if $A(v)$ denotes the label output by the algorithm, for a random node v,*

$$\lim_{p\to 0}\lim_{n\to\infty} \Pr[A(v) = \sigma_v] = \frac{1}{2} + \mathsf{T}^*(\tfrac{a+b}{2}, \tfrac{b}{a+b})$$

A.4 Better-than-random Reconstruction with Local Algorithms in the Labeled Model for Any k

Given an instance of the stochastic block model $(G, \sigma, R) \sim \mathcal{G}(n, k, a, b, p)$ and the corresponding Galton-Watson tree and broadcast process $(T, \tau, R) \sim \mathcal{T}(T, k, \eta, p)$, we now prove that if $d\lambda^2 = (a-b)^2/(k(a+(k-1)b)) > 1$, the plurality of labels at distance ℓ from a node v provides a robust way to recover a v's label for every information p. The argument is based on the reconstruction argument for the label of a root in a broadcast process on trees, and the fact that the application of the second moment method in this argument is robust to noise in the leaf labels. This was implicit in [25] and more explicit in [23]. Interestingly, the proof will show that in the case of Poisson Galton-Watson tree, a simple plurality style rule is sufficient for reconstruction.

▶ **Proposition 19.** *Let $(G, \sigma, R) \sim \mathcal{G}(n, k, a, b, p)$, with $a + (k-1)b > k$. Then, there exists a constant $\epsilon = \epsilon(a, b, k, p)$, such that if $(a-b)^2 > k(a+(k-1)b)$, there is a local algorithm A such that if $A(v)$ denotes the label output by the algorithm, for a random node v,*

$$\Pr[A(v) = \sigma_v] \geq \frac{1}{k} + \epsilon.$$

The result also holds for the noisy-label model.

The proof follows more or less directly from previous results [25], but we also provide it in the full version [16] for completeness.

B Conjecture

B.1 The Uselessness of Global Information

In the case of two clusters, we conjecture that whenever any node label information is present, a local algorithm is already able to recover the clusters optimally. The algorithm is the following: Fix some radius r, for each $v \in G$, look at the neighborhood $G_r(v)$, let $R_r \subseteq G_r(v)$ denote the revealed nodes in the neighborhood. As long as $r \leq c\log(n)$ for a sufficiently small constant c, the neighborhood is a tree with high probability. Then $\Pr[\sigma_v = 1 \mid R_r, \sigma_{R_r}]$ can be computed exactly by belief propagation. We conjecture that this is optimal. This would follow from a related conjecture regarding the broadcast process on trees and an application of Lemma 9.

▶ **Conjecture 20.** *Let T be infinite tree with root ρ. Let $(T, \tau, R) \sim \mathcal{T}(T, 2, \eta, p)$ (see Section 2). Then for any $p > 0$ and $\eta < 1/2$,*

$$\lim_{r\to\infty} \mathbb{E}\big|\Pr[\tau_{\rho=1} \mid \tau_R] - \Pr[\tau_{\rho=1} \mid \tau_R, \tau_{\partial T_r}]\big| = 0.$$

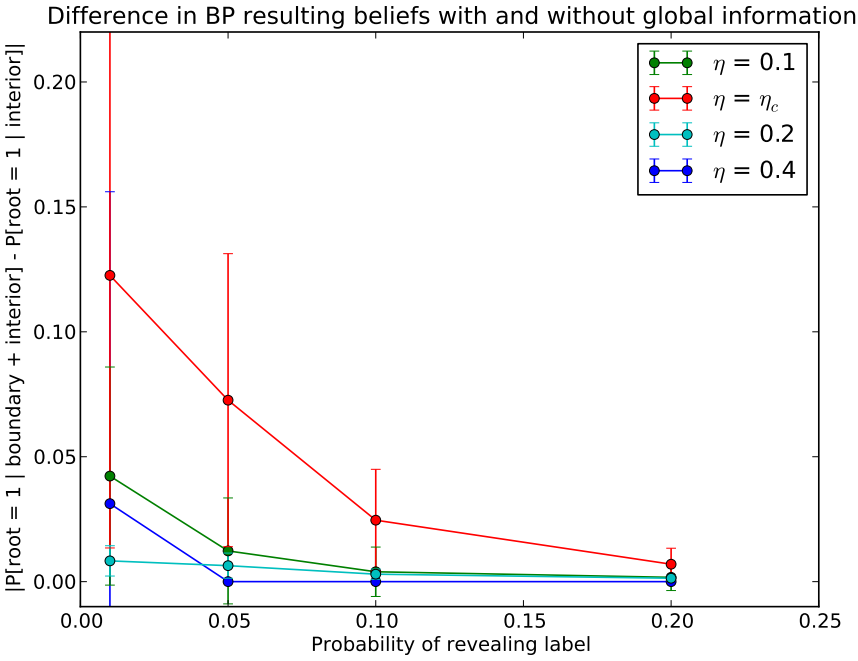


Figure 2 The average distance $|p_{R,L} - p_R|$ is shown for $\eta = 0.1, \eta_c, 0.3, 0.4$ and $p = 0.01, 0.05, 0.1, 0.2$.

B.2 Simulation

To test this conjecture, we ran the Belief Propagation algorithm on 3-regular trees of depth 10, in which labels were assigned to nodes according to broadcast processes starting at the root. Let L denote the set of leaves at level 10. Each node in the interior was revealed independently with probability p, to get the set R. We considered $p \in \{0.01, 0.05, 0.10, 0.20\}$. We also tried various settings of the broadcast parameter, η. We chose $\eta \in \{0.1, \eta_c, 0.3, 0.4\}$, where $\eta_c = \frac{1}{2}\left(1 - \frac{1}{\sqrt{3}}\right)$ is the threshold value for the setting considered.

The labeling process was always initiated with the root having label 1. Thus, we were interested in the posterior probability of the root being labeled 1 in various cases. We computed this posterior probability in three cases: (i) using only the labels at the leaves, denoted by p_L (ii) using only the interior nodes, denoted p_R, and (iii) using both the leaves and the interior nodes, denoted by $p_{L,R}$.

In the first case, only global information is used—*i. e.*, the set of labels at the boundary is the maximum possible information that can be inferred using the global properties of the graph. Thus, in some sense this is an upper bound on the utility of global information. In the second case, only local information in the form revealed nodes in the neighborhood is used. Finally, in the the third case, both local and global information is used.

Our conjecture suggests that as $r \to \infty$, $|p_{R,L} - p_R| \to 0$. Figure 2 shows our results. Each plot corresponds to a fixed value of η, and displays the average distance $|p_{R,L} - p_R|$ for different values of p. We ran the simulation multiple times for each setting of p and η and the standard deviation is marked on the plot.

Embedding Hard Learning Problems Into Gaussian Space

Adam Klivans and Pravesh Kothari

The University of Texas at Austin, Austin, Texas, USA
{klivans,kothari}@cs.utexas.edu

Abstract

We give the first representation-independent hardness result for agnostically learning halfspaces with respect to the Gaussian distribution. We reduce from the problem of learning sparse parities with noise with respect to the uniform distribution on the hypercube (sparse LPN), a notoriously hard problem in theoretical computer science and show that any algorithm for agnostically learning halfspaces requires $n^{\Omega(\log(1/\epsilon))}$ time under the assumption that k-sparse LPN requires $n^{\Omega(k)}$ time, ruling out a polynomial time algorithm for the problem. As far as we are aware, this is the first representation-independent hardness result for supervised learning when the underlying distribution is restricted to be a Gaussian.

We also show that the problem of agnostically learning sparse polynomials with respect to the Gaussian distribution in polynomial time is as hard as PAC learning DNFs on the uniform distribution in polynomial time. This complements the surprising result of Andoni et. al. [1] who show that sparse polynomials are learnable under *random* Gaussian noise in polynomial time.

Taken together, these results show the inherent difficulty of designing supervised learning algorithms in Euclidean space even in the presence of strong distributional assumptions. Our results use a novel embedding of random labeled examples from the uniform distribution on the Boolean hypercube into random labeled examples from the Gaussian distribution that allows us to relate the hardness of learning problems on two different domains and distributions.

1998 ACM Subject Classification F.2.0. Analysis of Algorithms and Problem Complexity

Keywords and phrases distribution-specific hardness of learning, gaussian space, halfspace-learning, agnostic learning

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.793

1 Introduction

Proving lower bounds for learning Boolean functions is a fundamental area of study in learning theory ([3, 14, 12, 31, 8, 26, 30, 10]). In this paper, we focus on representation-independent hardness results, where the learner can output any hypothesis as long as it is polynomial-time computable. Almost all previous work on representation-independent hardness induces distributions that are specifically tailored to an underlying cryptographic primitive and only rule out learning algorithms that succeed on all distributions.

Given the ubiquity of learning algorithms that have been developed in the presence of distributional constraints (e.g., margin-based methods of [2, 4, 35] and Fourier-based methods of [22, 27]), an important question is whether functions that seem difficult to learn with respect to all distributions are in fact also difficult to learn even with respect to natural distributions. In this paper we give the first hardness result for a natural learning problem (agnostically learning halfspaces) with respect to perhaps the strongest possible distributional constraint, namely that the marginal distribution is a spherical multivariate Gaussian.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 793–809

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

1.1 Learning Sparse Parities With Noise

Our main hardness result is based on the assumption of hardness of *learning sparse parities with noise*. Learning parities with noise (LPN) and its sparse variant are notoriously hard problems with connections to cryptography [25] in addition to several important problems in learning theory [13]. In this problem, the learner is given access to random examples drawn from the uniform distribution on the n dimensional hypercube (denoted by $\{-1,1\}^n$) that are labeled by an unknown parity function. Each label is flipped with a fixed probability η (*noise rate*), independently of others. The job of the learner is to recover the unknown parity. In the sparse variant, the learner is additionally promised that the unknown parity is on a subset of size at most a parameter k. It is easy to see that the exhaustive search algorithm for k-SLPN runs in time $n^{O(k)}$, and an outstanding open problem is to find algorithms that significantly improve upon this bound. The specific hardness assumption we take is as follows:

▶ **Assumption 1.** *Any algorithm for learning k-SLPN for any constant accuracy parameter ϵ must run in time $n^{\Omega(k)}$.*

The current best algorithm for SLPN is due to Greg Valiant [36] and runs in time $\Omega(n^{0.8k})$ for constant noise rates. Finding even an $O(n^{k/2})$-time algorithm for SLPN would be considered a breakthrough result. We note that the current best algorithm for LPN is due to Blum. et. al. [5] and runs in time $2^{O(n/\log n)}$.

Further evidence for the hardness of SLPN are the following surprising implications in learning theory: 1) an $n^{o(k)}$-time algorithm for SLPN would imply an $n^{o(k)}$-time algorithm for learning k-juntas and 2) a polynomial-time algorithm for $O(\log n)$-SLPN would imply a polynomial-time algorithm for PAC learning DNF formulas with respect to the uniform distribution on the cube *without queries* due to a reduction by Feldman et. al. [13]. The LPN and SLPN problems have also been used in previous work to show representation-independent hardness for agnostically learning halfspaces with respect to the uniform distribution on $\{-1,1\}^n$ [22] and for agnostically learning non-negative submodular functions [15].

1.2 Our results

We focus on giving hardness results for agnostically learning halfspaces and sparse polynomials. Learning halfspaces is one of the most well-studied problems in supervised learning. A halfspace (also known as a *linear classifier* or a *linear threshold function*) is a Boolean valued function (i.e. in $\{-1,1\}$) that can be represented as $\mathsf{sign}(\sum_{i=1}^{n} a_i \cdot x_i + c)$ for reals $a_1, a_2, \ldots, a_n$ and c with the input x being drawn from any fixed distribution on $\mathbb{R}^n$. Algorithms for learning halfspaces form the core of important machine learning tools such as the Perceptron [34], Artificial Neural Networks [38], Adaboost [18] and Support Vector Machines (SVMs) [38].

While halfspaces are efficiently learnable in the noiseless (PAC model of Valiant [37]) setting, the wide applicability of halfspace learning algorithms to labeled data that are not linearly separable has motivated the question of learning noisy halfspaces. Blum et. al. [6] gave an efficient algorithm to learn halfspaces under random classification noise. However, under adversarial noise (i.e.the *agnostic* setting), algorithmic progress has been possible only with distributional assumptions. Kalai et. al. [22] showed that halfspaces are agnostically learnable on the uniform distribution on the hypercube in time $n^{O(1/\epsilon^2)}$ and on the gaussian distribution in time $n^{O(1/\epsilon^4)}$. The latter running time was improved to $n^{O(1/\epsilon^2)}$ by Diakonikolas et. al. [9]. Shalev-Schwartz. et. al. [35] have given efficient agnostic algorithms for learning halfspaces in the presence of a large margin (their results

do not apply on spherical Gaussian distribution, as halfspaces with respect to Gaussian distributions may have exponentially small margins).

Kalai et. al. [22] showed that their agnostic learning algorithm on the uniform distribution on the hypercube is in fact optimal, assuming the hardness of the learning parity with noise (LPN) problem. No similar result, however, was known for the case of Gaussian distribution:

▶ **Question 1.** *Is there an algorithm running in time* $\mathsf{poly}(n, 1/\epsilon)$ *to agnostically learn halfspaces on the Gaussian distribution?*

There was some hope that perhaps agnostically learning halfspaces with respect to the Gaussian distribution would be easier than on the uniform distribution on the hypercube. We show that this is not the case and give a negative answer to the above question. In fact, we prove that any agnostic learning algorithm for the class of halfspaces must run in time $n^{\Omega(\log(1/\epsilon))}$.

▶ **Theorem 1** (See Theorem 8 for details)**.** *If Assumption 1 is true, any algorithm that agnostically learns halfspaces with respect to the Gaussian distribution to an error of* ϵ *runs in time* $n^{\Omega(\log(1/\epsilon))}$.

We next consider the problem of agnostically learning sparse (with respect to the number of monomials) polynomials. Since this is a real valued class of functions, we will work with the standard notion of ℓ_1 distance to measure errors. Thus, the distance between two functions f and g on the Gaussian distribution is given by $\mathbb{E}_{x\sim\gamma}[|f(x) - h(x)|]$. Note that ℓ_1 error reduces to the standard disagreement (or classification) distance in case of Boolean valued functions.

▶ **Question 2** (Agnostic Learning of Sparse Polynomials)**.** *For a function* $f : \mathbb{R}^n \to \mathbb{R}$*, normalized so that* $\mathbb{E}_{x\sim\gamma}[f(x)^2] = 1$*, suppose there is an* s*-sparse polynomial* p *such that* $\mathbb{E}_{x\sim\gamma}[|f(x) - p(x)|] \leq \delta \in [0, 1]$*. Is there an algorithm that uses random examples labeled by* f *to return a hypothesis* h *such that* $\mathbb{E}_{x\sim\gamma}[|h(x) - f(x)|] \leq \delta + \epsilon$*, in time* $\mathsf{poly}(s, n, 1/\epsilon)$*?*

On the uniform distribution on $\{-1, 1\}^n$, even the noiseless version (i.e. $\delta = 0$) of the question above is at least as hard as learning juntas. Indeed, a $\mathsf{poly}(s, n, 1/\epsilon)$ time algorithm for PAC learning s-sparse polynomials yields the optimal (up to polynomial factors) run time of $\mathsf{poly}(2^k, n, 1/\epsilon)$ for learning juntas. Agnostically learning sparse polynomials on the uniform distribution on $\{-1, 1\}^n$ is at least as hard as the problem of PAC learning DNFs with respect to the uniform distribution on $\{-1, 1\}^n$, a major open question in learning theory.

On the other hand, a surprising recent result by Andoni et. al.[1] shows that it is possible to learn sparse polynomials in the presence of *random* additive Gaussian noise with respect to the Gaussian distribution (as opposed to the agnostic setting where the noise is adversarial). Given the results of Andoni. etl. al. [1], a natural question is if the agnostic version of the question is any easier with respect to the Gaussian distribution. We give a negative answer to this question:

▶ **Theorem 2** (See Theorem 10 for details)**.** *If Assumption 1 is true, then, there is no algorithm running in time* $\mathsf{poly}(n, s, 2^d, 1/\epsilon)$ *to agnostically learn* s*-sparse degree* d *polynomials from random examples on the Gaussian distribution.*

A subroutine to find heavy Fourier coefficients of any function f on $\{-1, 1\}^n$ is an important primitive in learning algorithms and the problem happens to be just as hard as agnostic learning sparse polynomials described above. On the Gaussian distribution, Fourier-Transform based methods employ what is known as the *Hermite* transform [22, 28].

We show that the problem of finding heavy Hermite coefficients of a function on $\mathbb{R}^n$ from random examples is no easier than its analog on the cube. In particular, we give a reduction from the problem of PAC learning DNF formulas on the uniform distribution, to the problem of finding heavy Hermite coefficients of a function on $\mathbb{R}^n$ given random examples labeled by it. It is possible to derive this result by using the reduction of Feldman et. al.[13] who reduce PAC learning DNF formulas on the uniform distribution to sparse LPN by combining it with our reduction from sparse LPN to agnostic learning of sparse polynomials. However, we give a simple direct proof based on the properties of Fourier spectrum of DNF formulas due to [21].

To complement this negative result, we show that the problem becomes tractable if we are allowed the stronger value query access to the target function, in that, the learner can query any point of its choice and obtain the value of the target at the point from the oracle. On the uniform distribution on the hypercube, with query access to the target function, the task of agnostic learning s-sparse polynomials can in fact be performed in polynomial time in $s, n, 1/\epsilon$ using the well known Kushilevitz-Mansour (KM) algorithm [32]. The KM algorithm can be equivalently seen as a procedure to find the large Fourier coefficients of a function given query access to it. We show (in Appendix A) that it is possible to extend the KM algorithm to succeed in finding heavy Hermite coefficients.

▶ **Theorem 3.** *Given access to a queries from a function f such that $\mathbb{E}_{x\sim\gamma}[f(x)^2] = 1$, there is an algorithm that finds all the Hermite coefficients of f of degree d that are larger in magnitude than ϵ, in time $\mathsf{poly}(n, d, 1/\epsilon)$. Consequently, there exists an algorithm to agnostically learn s-sparse degree d polynomials on γ in time and queries $\mathsf{poly}(s, n, d, 1/\epsilon)$.*

1.3 Our Techniques

Our main result relates the hardness of agnostic learning of halfspaces on the Gaussian distribution to the hardness of learning sparse parities with noise on the uniform distribution on the hypercube (sparse LPN). The reduction involves embedding a set of labeled random examples on the hypercube into a set of labeled random examples on $\mathbb{R}^n$ such that the marginal distribution induced on $\mathbb{R}^n$ is the Gaussian distribution. To do this, we define an operation that we call as the *Gaussian lift* of a function, that takes an example label pair $(x, f(x))$ with $x \in \{-1, 1\}^n$ and produces $(z, f^\gamma(z))$ where z is distributed according to the Gaussian distribution if x is distributed according to the uniform distribution on $\{-1, 1\}^n$. We refer to the function f^γ as the *Gaussian lift* of f.

We show that given random examples labeled by f from the uniform distribution on $\{-1, 1\}^n$, one can generate random examples labeled by f^γ whose marginal distribution is the Gaussian. Further, we show how to recover a hypothesis close to f from a hypothesis close to f^γ. When f is a parity function, f^γ will be noticeably correlated with some halfspace. We show that the correlation is in fact exponentially small in n (but still enough to give us our hardness results) and requires a delicate computation which we accomplish looking at it as a limit of a quantity that can be estimated accurately. We then implement a similar idea for reducing sparse LPN to agnostically learning sparse polynomials on $\mathbb{R}^n$ under the Gaussian distribution by proving that the Gaussian lift of the parity function χ^γ is correlated with a monomial on $\mathbb{R}^n$ with respect to the Gaussian distribution.

We note that when allowed query access to the target function on $\{-1, 1\}^n$, one can extend the well known KM algorithm [32] to find heavy Hermite coefficients of any function on $\mathbb{R}^n$, given query access to it. The main difference in this setting is the presence of higher degree terms in Hermite expansion (as against only multilinear terms in the Fourier expansion).

1.4 Related Work

We survey some algorithms and lower bounds for the problem of agnostically learning halfspaces here. As mentioned before, [22], gave agnostic learning algorithms for halfspaces by assuming that the distribution is product Gaussian. They showed that their algorithm can be made to work under the more challenging log-concave distributions in polynomial time for any constant error. This result was recently improved by [23]. [35] gave a polynomial time algorithm for the problem under *large margin assumptions* on the underlying distribution. Following this, [4] gave a trade-off between time and accuracy in the large margin framework.

In addition to the representation-independent hardness results mentioned before, there is a line of work that shows *proper* hardness of agnostically learning halfspaces on arbitrary distributions via a reduction from hard problems in combinatorial optimization. [19] show that it is NP hard to properly (i.e. the hypothesis is restricted to be a halfspace) agnostically learn halfspaces on arbitrary distributions. Extending this result, [11] show that it is impossible to give an agnostic learning algorithm for halfspaces on arbitrary distribution that returns a polynomial threshold function of degree 2 as the hypothesis, unless P = NP.

2 Preliminaries

In this paper, we will work with functions that take both real and Boolean values (i.e. in $\{-1,1\}$) on the n-dimensional hypercube $\{-1,1\}^n$ and $\mathbb{R}^n$. For an element $x \in \{-1,1\}^n$, we will denote the coordinates of x by x_i. Let $\gamma = \gamma_n$ be the standard product Gaussian distribution on $\mathbb{R}^n$ with mean 0 and variance 1 in every direction and $\mathcal{U} = \mathcal{U}_n$, the uniform distribution on $\{-1,1\}^n$. We define the *sign* function on $\mathbb{R}$ as $\mathsf{sign}(x) = x/|x|$ for every $x \neq 0$. Set $\mathsf{sign}(0)$ to be 0. For $z \in \{-1,1\}^n$, the weight of z is the translated Hamming weight (to account for our bits being in $\{-1,1\}$) and is denoted by $|z| = \frac{1}{2}\sum_{i \in n} z_i + n/2$. For vectors $z \in \{-1,1\}^n$ and $y \in \mathbb{R}^n_+$, let $z \circ y$ denote the vector t such that $t_i = z_i \cdot y_i$.

A *half normal* random variable is distributed as the absolute value of a univariate gaussian random variable with mean zero and variance 1. We denote the distribution of a half normal random variable by $|\gamma|$. As is well known, $\mathbb{E}_{x \sim |\gamma|}[x] = \sqrt{2/\pi}$ and $Var[|\gamma|] = (1 - 2/\pi)$.

The *parity* function $\chi_S : \{-1,1\}^n \to \{-1,1\}$, for any $S \subseteq [n]$, is defined by $\chi_S(x) = \Pi_{i \in S} x_i$ for any $x \in \{-1,1\}^n$. For any $S \subseteq [n]$, the *majority* function MAJ_S is defined by $Maj_S(x) = \mathsf{sign}(\sum_{i \in S} x_i)$. The input x in the current context will come either from $\{-1,1\}^n$ or $\mathbb{R}^n$. When $S = [n]$, we will drop the subscript and write χ and MAJ for $\chi_{[n]}$ and $\mathsf{MAJ}_{[n]}$ respectively. The class of halfspaces is the class of all Boolean valued functions computed by expressions of the form $\mathsf{sign}(\sum_{i \in [n]} a_i \cdot x_i)$ for coefficients $a_i \in \mathbb{R}$ for each $1 \leq i \leq n$. The inputs to a halfspace can come from both $\{-1,1\}^n$ and $\mathbb{R}^n$.

For a probability distribution $\mathcal{D}$ on X ($\mathbb{R}^n$ or $\{-1,1\}^n$) and any functions $f, g : X \to \mathbb{R}$ such that $\mathbb{E}_{x \sim \mathcal{D}}[f(x)^2], \mathbb{E}_{x \sim \mathcal{D}}[g(x)^2] < \infty$, let $\langle f, g \rangle_{\mathcal{D}} = \mathbb{E}_{x \sim \mathcal{D}}[f(x) \cdot g(x)]$. The ℓ_1 and ℓ_2 norms of f w.r.t $\mathcal{D}$ are defined by $||f||_1 = \mathbb{E}_{x \sim \mathcal{D}}[|f(x)|]$ and $||f||_2 = \sqrt{\mathbb{E}_{x \sim \mathcal{D}}[f(x)^2]}$, respectively. We will drop the subscript in the notation for inner products when the underlying distribution is clear from the context.

Fourier Analysis on $\{-1,1\}^n$: Parity functions for each $\alpha \subseteq [n]$ form an orthonormal basis for the linear space of all real valued square summable functions on the uniform distribution on $\{-1,1\}^n$ (denoted by $L^2(\{-1,1\}^n, \mathcal{U})$). The (real) coefficients of the linear combination are referred to as the Fourier coefficients of f. For $f : \{-1,1\}^n \to \mathbb{R}$ and $\alpha \subseteq [n]$, the *Fourier coefficient* $\hat{f}(\alpha)$ is given by $\hat{f}(\alpha) = \langle f, \chi_\alpha \rangle = \mathbb{E}[f(x)\chi_\alpha(x)]$. The cardinality of the index set α is said to be the *degree* of the Fourier coefficient $\hat{f}(\alpha)$. The *Fourier*

expansion of f is given by $f(x) = \sum_{\alpha \subseteq [n]} \hat{f}(\alpha)\chi_\alpha(x)$. Finally, we have *Plancherel's theorem*: $\langle f, g\rangle_{\mathcal{U}} = \sum_{\alpha\subseteq[n]} \hat{f}(\alpha)\cdot\hat{g}(\alpha)$ and $||f||_2^2 = \sum_{\alpha\subseteq[n]}\hat{f}(\alpha)^2$ for any $f \in L^2(\{-1,1\}^n, \mathcal{U})$.

It is possible to exactly compute the Fourier coefficients of the Majority function $\mathsf{MAJ}_{[n]} = \mathsf{MAJ}$ on $\{-1,1\}^n$. We refer the reader to the online lecture notes of O'Donnell (Theorem 16, [33]).

▶ **Fact 1.** *Let* $\widehat{\mathsf{MAJ}}(\alpha)$ *be the Fourier coefficients of the majority function* $\mathsf{MAJ} = \mathsf{MAJ}_{[n]}$ *at index set* α *of cardinality* a *on* $\{-1,1\}^n$ *for an odd* n*. As* MAJ *is a symmetric function, the values of the coefficients depend only on the cardinality of the index set* a*. As* MAJ *is an odd function,* $\widehat{\mathsf{MAJ}}(\alpha) = 0$ *if* $|\alpha| = a$ *is even. For odd* a*:*

$$\widehat{\mathsf{MAJ}}(\alpha) = (-1)^{\frac{a-1}{2}} \cdot \frac{\binom{\frac{n-1}{2}}{\frac{a-1}{2}}}{\binom{n-1}{a-1}} \cdot \frac{2}{2^n} \cdot \binom{n-1}{\frac{n-1}{2}}.$$

In particular, $\widehat{\mathsf{MAJ}}(\alpha) = \sqrt{\frac{2}{n\pi}}$ *if* $a = 1$*.*

Hermite Analysis on $\mathbb{R}^n$: Analogous to the parity functions, the *Hermite polynomials* form an orthonormal and complete basis for $L^2(\mathbb{R}^n, \gamma_n)$, the linear space of all square integrable functions on $\mathbb{R}^n$ with respect to the spherical gaussian distribution $\gamma_n = \gamma$. These polynomials can be constructed in the univariate ($n = 1$) case by applying Gram Schmidt process to the family $\{1, x, x^2, \ldots\}$ giving the first few members as $h_0(x) = 1$, $h_1(x) = x$, $h_2(x) = \frac{x^2-1}{\sqrt{2}}$, $h_3(x) = \frac{x^3-3x}{\sqrt{6}}$, $\cdots$. The multivariate Hermite polynomials are obtained by taking products of univariate Hermite polynomials in each coordinate. Thus, for every n-tuple of non-negative integers $\Delta = (d_1, d_2, \ldots, d_n) \in \mathbb{Z}^n$, we have a polynomial $H_\Delta = \Pi_{i\in[n]} h_{d_i}(x_i)$. As γ_n is product and h_{d_i} are each orthonormal, H_Δ so constructed are clearly an orthonormal family of polynomials.

Analogous to the Fourier expansion, any function $f \in L^2(\mathbb{R}^n, \gamma_n)$ can be written uniquely as $\sum_{D\in\mathbb{Z}^n} \hat{f}(\Delta)\cdot H_\Delta$, where $\hat{f}(\Delta)$ is the *Hermite coefficient* of f at index Δ and is given by $\hat{f}(\Delta) = \mathbb{E}_{x\sim\gamma}[f(x)\cdot H_\Delta(x)]$. We have the Plancherel's theorem: $\langle f, g\rangle_\gamma = \sum_{\Delta\subseteq\mathbb{Z}^n} \hat{f}(\Delta)\cdot\hat{g}(\Delta)$ and $||f||_2^2 = \sum_{\Delta\subseteq\mathbb{Z}^n}\hat{f}(\Delta)^2$ for any $f \in L^2(\mathbb{R}^n, \gamma)$.

Agnostic Learning: The agnostic model of learning [20, 24] is a challenging generalization Valiant's PAC model of supervised learning that allows adversarial noise in the labeled examples. Given labeled examples from an arbitrary target function p, the job of an agnostic learner for a class $\mathcal{C}$ of real (or Boolean) valued functions is to produce a hypothesis h that has an error w.r.t p that is at most ϵ more than that of best fitting hypothesis from the class $\mathcal{C}$. Formally, we have:

▶ **Definition 4** (Agnostic learning with ℓ_1 error)**.** Let $\mathbb{F}$ be a class of real-valued functions with distribution $\mathcal{D}$ on X (either $\{-1,1\}^n$ or $\mathbb{R}^n$). For any real valued target function p on X, let $\mathrm{opt}(p, \mathbb{F}) = \inf_{f\in\mathbb{F}} \mathbb{E}_{x\sim\mathcal{D}}[|p(x) - f(x)|]$. An algorithm $\mathcal{A}$, is said to agnostically learn $\mathbb{F}$ on $\mathcal{D}$ if for every $\epsilon > 0$ and any target function p on X, given access to random examples drawn from $\mathcal{D}$ and labeled by p, with probability at least $\frac{2}{3}$, $\mathcal{A}$ outputs a hypothesis h such that $\mathbb{E}_{(x,y)\sim\mathsf{P}}[|h(x) - p(x)|] \leq \mathrm{opt}(p, \mathbb{F}) + \epsilon$.

The ℓ_1 error for real valued functions specializes to the disagreement (or Hamming) error for Boolean valued functions and thus the definition above is a generalization of agnostic learning a class of Boolean valued functions on a distribution. A general technique (due to [22]) for agnostic learning $\mathcal{C}$ on any distribution $\mathcal{D}$ is to show that every function in $\mathcal{C}$ is

approximated up to an ℓ_1 error of at most ϵ by a polynomial of low-degree $d(n, \epsilon)$, which can then be constructed using ℓ_1-polynomial regression. This approach to learning can equivalently be seen as learning based on Empirical Risk Minimization with absolute loss [38]. As observed (in [22]), since ℓ_1 error for Boolean valued functions is equivalent to the disagreement error, polynomial regression can also be used to agnostically learn Boolean valued function classes w.r.t disagreement error.

3 Hardness of Agnostically Learning Halfspaces on the Gaussian Distribution

In this section, we show that any algorithm that agnostically learns the class of halfspaces on the Gaussian distribution with an error of at most ϵ takes time $n^{\Omega(\log(1/\epsilon))}$. In particular, there is no fully polynomial time algorithm to agnostically learn halfspaces on the Gaussian distribution (subject to the hardness of sparse LPN). We reduce the problem of learning sparse parities with noise on the uniform distribution on the Boolean hypercube to the problem of agnostic learning halfspaces on the Gaussian distribution to obtain our hardness result.

Our approach is a generalization of the one adopted by [22] who used such a reduction to show the optimality of their agnostic learning algorithm for halfspaces on the uniform distribution on $\{-1, 1\}^n$. We begin by briefly recalling their idea here: Let χ_S be the unknown parity for some $S \subseteq [n]$. Observe that on the uniform distribution on $\{-1, 1\}^n$, χ_S is correlated with the majority function MAJ_S with a correlation of $\approx 1/\sqrt{|S|} \geq 1/\sqrt{n}$. Thus, the expected correlation between MAJ_S and the noisy labels is $\approx \eta/\sqrt{n}$ where η is the noise rate. In other words, MAJ_S predicts the value of the label at a uniformly random points from $\{-1, 1\}^n$ with probability $\approx 1/2 + \eta/\sqrt{n}$ (i.e. with an inverse polynomial advantage over random). The key idea here is to note that if we drop a coordinate, say $j \in S$ (i.e. a "relevant" variable for the unknown parity) from every example point to obtain labeled examples from $\{-1, 1\}^{n-1}$, then, the labels and example points are independent as random variables and thus no halfspace can predict the labels to an inverse polynomial advantage. On the other hand, if we drop a coordinate $j \notin S$, then, the labels are still correlated with the correct parity and thus, MAJ_S predicts the labels with an inverse polynomial advantage. Thus, drawing enough examples can allow us to distinguish between the two cases and construct S one variable at a time.

Such a strategy, however, cannot be directly applied to relate learning problems on *different* distributions. Instead, we show that given examples from $\{-1, 1\}^n$ labeled by some function f, we can simulate examples drawn according to the Gaussian distribution, labeled by some $f^\gamma : \mathbb{R}^n \to \{-1, 1\}$ (which we call as the *Gaussian lift* of f). Further, we show that when f is some parity χ_α, then, f^γ is noticeably correlated with some halfspace on the Gaussian distribution. Now, given examples drawn according to the Gaussian distribution, labeled by some f^γ, one can use the agnostic learner for halfspaces to recover α with high probability. We now proceed with the details of our proof. We first define the Gaussian lift of any function $f : \{-1, 1\}^n \to \mathbb{R}$. At any $x \in \mathbb{R}^n$, f^γ returns a value obtained by evaluating f at the point associated with the sign pattern of x.

▶ **Definition 5** (Gaussian Lift). The Gaussian lift of a function $f : \{-1, 1\}^n \to \mathbb{R}$ is a function $f^\gamma : \mathbb{R}^n \to \mathbb{R}$ such that for any $x \in \mathbb{R}^n$, $f^\gamma(x) = f(\mathsf{sign}(x_1), \mathsf{sign}(x_2), \ldots, \mathsf{sign}(x_n))$.

We begin with a general reduction from the problem of learning k-sparse parity with noise on the uniform distribution on $\{-1, 1\}^n$ to problem of learning any class $\mathcal{C}$ of functions

on $\mathbb{R}^n$ agnostically on the Gaussian distribution. This reduction works under the assumption that for every $S \subseteq [n]$, the Gaussian lift of the parity function on $\alpha \subseteq [n]$, denoted as χ_α^γ is noticeably correlated with some function from $c_\alpha \in \mathcal{C}$.

▶ **Lemma 6** (Correlation Lower Bound yields Reduction to SLPN). *Let $\mathcal{C}$ be a class of Boolean valued functions on $\mathbb{R}^n$ such that for every $\alpha \subseteq [n]$ for $|\alpha| \leq k$, there exists a function $c_\alpha \in \mathcal{C}$ such that $\langle c_\alpha, \chi_\alpha^\gamma \rangle \geq \theta(k)$and c_α depends on variables only in α.*

Suppose there exists an algorithm $\mathcal{A}$ (that may not be proper *and can output real valued hypotheses) to learn $\mathcal{C}$ agnostically over the Gaussian distribution to an ℓ_1 error of at most ϵ using time and samples $T(n, 1/\epsilon)$. Then, there exists an algorithm to solve SLPN that runs in time and examples $\tilde{O}(\frac{n}{(1-2\eta)\theta(k)}) + \tilde{O}(n) \cdot T(n, \frac{2}{(1-2\eta)\theta(k)})$ where η is the noise rate.*

Proof. We will assume that $\mathcal{C}$ is negation closed, that is, for every $c \in \mathcal{C}$, the function $-c \in \mathcal{C}$. This assumption can be easily removed by running the procedure described below twice, the second time with the labels of the examples negated. We skip the details of this easy adjustment here. Let χ_β be the target parity for some $\beta \subseteq [n]$ such that $|\beta| \leq k$. We claim that the following procedure determines if $j \in \beta$ for any $j \in [n]$ given noisy examples from χ_β with high probability.

1. For each example-label pair $(x, y) \in \{-1, 1\}^n \times \{-1, 1\}$, generate a new example label pair as follows.
   a. Draw independent half-normals $h_1, h_2, ..., h_n$.
   b. Let $z \in \mathbb{R}^{n-1}$ be defined so that $z_i = x_i \cdot h_i$ for each $i \in [n]$, $i \neq j$.
   c. Output (z, y) where $z = (z_1, z_2, \cdots z_{j-1}, z_{j+1}, \cdots, z_n) \in \mathbb{R}^{n-1}$. Denote the distribution of (z, y) by $\mathcal{D}_j$.
2. Set $\epsilon = (1 - 2\eta) \cdot \theta(k)$. Collect a set of $T(n, 1/\epsilon)$ examples, R, output by the procedure above.
3. Run $\mathcal{A}$ on R with ℓ_1 error parameter ϵ set to $(1 - 2\eta)\theta(k)/2$. Let h be the output of the algorithm.
4. Draw a set fresh set of $r = O(\log(1/\delta)/\epsilon^2)$, $\{(z^1, y^1), (z^2, y^2), \ldots, (z^r, y^r)\}$, again by the procedure above and estimate $err = \frac{1}{r} \cdot \sum_{i=1}^{r} [|h(z^i) - y^i|]$. Accept i as relevant if $err \leq 1 - \epsilon/4$. Else reject.

We now argue the correctness of this procedure. For D_j described above (obtained by dropping the j^{th} coordinate in the lifted examples), it is easy to see that the marginal distribution on the first $n - 1$ coordinates is γ_{n-1}, the spherical Gaussian distribution on $n - 1$ variables. Set $\epsilon = (1 - 2\eta) \cdot \theta(k)$.

Suppose $j \notin T$. In this case, for any example (z, y), $y = \chi_\beta^\gamma(z)$ with probability $1 - \eta$ independently of other examples. We know that there exists a $c_\beta \in \mathcal{C}$, depending only on coordinates in β such that $\langle c_\beta, \chi_\beta^\gamma \rangle \geq \theta(k)$. Thus, $\mathbb{E}_{(z,y)\sim\mathcal{D}_j}[c_\beta(z) \cdot y] = \epsilon$. In this case, thus, running $\mathcal{A}$ with the error parameter ϵ obtains $h : \{-1, 1\}^{n-1} \to \mathbb{R}$ with error at most

$$\begin{aligned}\mathbb{E}_{(z,y)\sim\mathcal{D}_j}[|c_\beta - y|] + \epsilon &= (1 - \eta) \cdot \mathbb{E}_{(z,y)\sim\mathcal{D}_j}[|c_\beta(z') - \chi_\beta^\gamma(z')|] + \\ &\qquad \eta \cdot \mathbb{E}_{(z,y)\sim\mathcal{D}_j}[|c_\beta(z') - \chi_\beta^\gamma(z')|]] + \epsilon \\ &= 1 - \epsilon/2.\end{aligned}$$

On the other hand, if $j \in T$, then, since the procedure drops the j^{th} coordinate of every example, the distribution of the labels y is uniformly random and independent of the distribution of the coordinates z_i, $i \neq j$. In this case, for any function in $h : \{-1, 1\}^{n-1} \to \mathbb{R}$, it can be easily checked that $\mathbb{E}[|c(z) - y|] \geq 1$ where the expectation is over the random variables (z, y).

We can estimate the ℓ_1 error $\mathbb{E}_{(z,y)\sim\mathcal{D}_j}[|y-h(z)|]$ of the hypothesis h produced by the algorithm, to an accuracy of $\epsilon/4$ with confidence $1-\delta$ using $r = O(\frac{\log(1/\delta)}{(1-2\eta)\theta(k)})$ examples. This is enough to distinguish between the two cases above. We can now repeat this procedure n times, once for every coordinate $j \in [n]$. Using a union bound, all random estimations and runs of $\mathcal{A}$ are successful with probability at least $2/3$ using an additional poly-logarithmic cost in n in time and samples required. Thus we obtain the stated running time and sample complexity. ◀

Next, we will show that χ_S^γ, the Gaussian lift of the parity function the subset $S \subseteq [n]$ is noticeably correlated with the majority function $\mathsf{MAJ}_S = \mathsf{sign}(\sum_{i\in S} x_i)$ with respect to the Gaussian distribution on $\mathbb{R}^n$. This correlation, while enough to yield the hardness result for agnostic learning of halfspaces when combined with Lemma 6, is an exponentially small quantity, in sharp contrast to the correlation between MAJ_S and χ_S on the uniform distribution on the hypercube (where it is $\approx 1/\sqrt{|S|}$). We thus need to adopt a more delicate method of estimating it as a limit of a quantity we can estimate accurately.

▶ **Lemma 7.** *Let m be an odd integer and consider $S \subseteq [n]$ such that $|S| = m$. Then,*

$$|\langle \mathsf{MAJ}_S, \chi_S^\gamma \rangle_\gamma| = 2^{-\Theta(m)}$$

Proof. Let $c = |\mathbb{E}_{x\sim\gamma_n}[\mathsf{MAJ}_S(x)\cdot\chi_S^\gamma(x)]|$. Each x_i above is independently distributed as $\mathcal{N}(0,1)$. Fix any odd integer t and define y_{ij} for each $1 \le i \le m$ and $1 \le j \le t$ to be uniform and independent random variables taking values in $\{-1,1\}$. The idea is to simulate each x_i by $\frac{1}{t}\sum_{j=1}^t y_{ij}$. In the limit as $t \to \infty$, the simulated random variable converges to in distribution to x_i. Call $f^t(x) = \mathsf{sign}(\sum_{i=1}^m \sum_{j=1}^t y_{ij})$ and $g^t(x) = \mathsf{sign}(\Pi_{i=1}^m \sum_{j=1}^t y_{ij})$, the functions obtained by applying the substitution above to MAJ_S and χ_S^γ respectively. Let $y = \{y_{ij} \in [m]\times[t]\}$ denote the inputs bits to f^t and g^t defined above. Thus,

$$c = \lim_{t\to\infty} \mathbb{E}[f^t \cdot g^t] = \lim_{t\to\infty} \mathbb{E}[\mathsf{sign}(\sum_{i=1}^m \sum_{j=1}^t y_{ij}) \cdot \Pi_{i=1}^m \mathsf{sign}(\sum_{j=1}^t y_{ij})] \tag{1}$$

Using Plancherel's Identity for the RHS above, we have:

$$c = \mathbb{E}[f^t \cdot g^t] = \sum_{\alpha\subseteq[m]\times[t]} \widehat{f^t}(\alpha)\cdot\widehat{g^t}(\alpha). \tag{2}$$

We now intend to estimate the RHS of the equation above. Towards this goal, we make some observations regarding the fourier coefficients $\widehat{f^t}(\alpha)$ and $\widehat{g^t}(\alpha)$.

▶ **Claim 1** (Fourier Coefficients of g^t). *For every $\alpha = \cup_{i=1}^m \alpha_i$ where $\alpha_i = \alpha \cap \{(i,j)|j\in[t]\}$ for each $1 \le i \le m$, $\widehat{g^t}(\alpha) = \Pi_{i=1}^m \widehat{\mathsf{MAJ}_{i\times[t]}}(\alpha_i)$.*

That is, the Fourier coefficient at α of g^t is the product of Fourier coefficients of majority functions at α_i, where the i^{th} majority function is on bits y_{ij} for $j \in [t]$.

Proof. $\widehat{g^t}(T) = \mathbb{E}[g^t(y)\cdot\chi_\alpha(y)] = \mathbb{E}[\Pi_{i=1}^m \mathsf{sign}(\sum_{j=1}^t y_{ij})\cdot\chi_\alpha(y) = \mathbb{E}[\Pi_{i=1}^m \chi_{\alpha_i}(y)\cdot\mathsf{sign}(\sum_{j=1}^t y_{ij})]$ $= \Pi_{i=1}^m \mathbb{E}[\mathsf{sign}(\sum_{j=1}^t y_{ij})\cdot\chi_{\alpha_i}(y)] = \Pi_{i=1}^m \widehat{\mathsf{MAJ}_{i\times[t]}}(\alpha_i)$, where for the third equality, we note that $\chi_\alpha = \Pi_{i=1}^m \chi_{\alpha_i}$ and for the last equality, the fact that y_{ij} are all independent and that α_i are disjoint. ◀

We now observe the term corresponding to each index α contributes a value with the same sign to the RHS of Equation (2).

▶ **Claim 2.** *Let $\alpha \subseteq [m] \times [t]$ and suppose $\widehat{f^t}(\alpha) \cdot \widehat{g^t}(\alpha) \neq 0$. If $m = 4q+1$ for $q \in \mathbb{N}$, then, $\mathsf{sign}(\widehat{f^t}(\alpha) \cdot \widehat{g^t}(\alpha)) = 1$. If $m = 4q+3$ for $q \in \mathbb{N}$ then, $\mathsf{sign}(\widehat{f^t}(\alpha) \cdot \widehat{g^t}(\alpha)) = -1$.*

Proof of Claim. Set $m = 4q+1$, the other case is similar. Recall that t is odd. Let $|\alpha| = a$ for some odd a (otherwise at least one of α_i is even in which case $\hat{g^t}(\alpha) = 0$). From Fact 1: $\mathsf{sign}(\widehat{f^t}(\alpha)) = (-1)^{(a-1)/2}$.

Let $\alpha = \cup_{i=1}^{m} \alpha_i$ such that $\alpha_i \subseteq [m] \times [t]$ and let for each i, $|\alpha_i| = a_i$. Using the claim above, we have: $\mathsf{sign}(\widehat{g^t}(\alpha)) = \Pi_{i=1}^{m} \mathsf{sign}(\widehat{\mathsf{MAJ}_{i\times[t]}}(\alpha_i)) = \Pi_{i=1}^{m}(-1)^{(a_i-1)/2} = (-1)^{(a-m)/2}$. Thus, $\mathsf{sign}(\widehat{f^t}(\alpha) \cdot \widehat{g^t}(\alpha)) = (-1)^{(2a-m-1)/2} = 1$. ◀

For the rest of the proof, assume that $m = 4q+1$. We are now in a position to analyze Equation (2). By Claim 2 above, we know that every term in the summation on the RHS of Equation (2) contributes a non-negative value. We group the Fourier coefficients of f and g based on the size of the index set and refer to the coefficients with index sets of size r by *layer* r. Observe that for any index set $\alpha \subseteq [m] \times [t] = \cup_{1 \leq i \leq m} \alpha_i$, if there is an i such that $\alpha_i = \emptyset$, then, $\widehat{g^t}(T) = 0$. Thus, the term corresponding to index α contributes 0 to the RHS of Equation (2). Thus, we can assume $|\alpha| \geq m$. We first estimate the contribution due to layer m:

▶ **Claim 3** (Contribution due to layer m)**.** *For large enough t,*

$$|\sum_{|\alpha|=m} \widehat{f^t}(\alpha) \cdot \widehat{g^t}(\alpha)| = \Omega\left(1/\sqrt{m} \cdot (\frac{2}{\pi e})^{m/2}\right).$$

Proof of Claim. Recall that $\alpha = \cup_{i=1}^{m} \alpha_i$ with each $\alpha_i \subseteq i \times [t]$. By the discussion above, $|\alpha_i| = 1$. There are exactly t^m indices α that satisfy this condition.

Using Fact 1 we know that $\hat{f^t}(\alpha) = (-1)^{\frac{t-1}{2}} \cdot \frac{\binom{\frac{tm-1}{2}}{\frac{m-1}{2}}}{\binom{tm-1}{m-1}} \cdot \frac{2}{2^{tn}} \cdot \binom{tm-1}{\frac{tm-1}{2}}$. Using Fact 1 again, for g^t along with Claim 1, we have : $\hat{g^t}(T) = \left(\sqrt{\frac{2}{t\pi}}\right)^m$. Thus, each non-zero term in layer m of (2) contributes: $\widehat{f^t}(\alpha) \cdot \widehat{g^t}(\alpha) = \frac{\binom{\frac{tm-1}{2}}{\frac{m-1}{2}}}{\binom{tm-1}{m-1}} \cdot \frac{2}{2^{tm}} \cdot \binom{tm-1}{\frac{tm-1}{2}} \cdot \left(\sqrt{\frac{2}{t\pi}}\right)^m$. Using asymptotically tight approximations for binomial coefficients, for large enough t:
$\frac{2}{2^{tm}} \cdot \binom{tm-1}{\frac{tm-1}{2}} = \Theta(\sqrt{\frac{1}{\pi \cdot (tm-1)}})$, and $\frac{\binom{\frac{tm-1}{2}}{\frac{m-1}{2}}}{\binom{tm-1}{m-1}} = \Omega((et)^{-\frac{m-1}{2}})$. Thus, the contribution to the RHS of Equation 2 by layer m asymptotically $\sum_{\alpha:|\alpha_i|=1} \widehat{f^t}(\alpha) \cdot \widehat{g^t}(\alpha) = \Omega(t^m \cdot t^{-\frac{m-1}{2}} \cdot e^{-\frac{m-1}{2}} \cdot \sqrt{\frac{2}{\pi \cdot (tm-1)}} \cdot \left(\sqrt{\frac{2}{t\pi}}\right)^m = \Omega\left(\frac{1}{\sqrt{m}} \cdot (\frac{2}{\pi e})^{m/2}\right)$. ◀

The claim above is enough to give us a lower bound on c. Our aim in the following is to establish an inverse exponential upper bound on the correlation between MAJ_S and χ_S^γ. Together with the contribution due to layer m, we have that $c = 2^{-\Theta(m)}$. This will complete the proof.

▶ **Claim 4** (Contribution due to layers $r > m$)**.** *For large enough t, m,*

$$|\sum_{|\alpha|>m} \widehat{f^t}(\alpha) \cdot \widehat{g^t}(\alpha)| = 2^{-\Omega(m)}.$$

Proof of Claim. Let $r_i \geq 1$ for every $1 \leq i \leq m$ such that $\sum_{i \leq n} r_i = r$. Consider any $\alpha = \cup_{1 \leq i \leq m} \alpha_i$ such that $|\alpha_i| = r_i$. The number of indices α is $\Pi_{1 \leq i \leq m} \binom{t}{r_i} \leq t^r / \Pi_{i=1}^{n} r_i!$. If

any r_i is even, then, the coefficient $\widehat{g^t}(\alpha) = 0$. Thus, the only non-zero contribution to the correlation from layer r is due to the indices α such that all $|\alpha_i| = r_i$ are odd positive integers. Using Fact 1: $\widehat{f^t}(\alpha) = \frac{\binom{\frac{tm-1}{2}}{\frac{r-1}{2}}}{\binom{tm-1}{r-1}} \cdot \frac{1}{2^{tm-1}} \cdot \binom{tm-1}{\frac{tm-1}{2}}$, and $|\widehat{g^t}(\alpha)| \leq \Pi_{1\leq i\leq m} \frac{\binom{\frac{t-1}{2}}{\frac{r_i-1}{2}}}{\binom{t-1}{r_i-1}} \frac{1}{2^{t-1}} \cdot \binom{t-1}{\frac{t-1}{2}}$.

Let us estimate the sum squared of all coefficients $\widehat{g^t}(\alpha)$ such that $|\alpha_i| = r_i$ for each i. Recall that for the majority function on m bits, the sum squared of all coefficients of any layer q is $\approx (2/\pi)^{3/2} \cdot 1/q^{3/2}$. This can be derived directly using Fact 1 (see [33]).

$$\sum_{\alpha:|\alpha_i|=r_i} (\widehat{g^t}(\alpha))^2 \leq \sum_{\alpha:|\alpha_i|=r_i} \Pi_{i=1}^m (\sum_{|\alpha_i|=r_i} (\widehat{\mathsf{MAJ}_{i\times[t]}}(\alpha_i))^2) = \Pi_{i=1}^m (2/\pi)^{3/2} \cdot 1/r_i^{3/2}.$$

The maximum value over all $r_1, r_2, \ldots, r_m$ that give a non-zero $\widehat{g^t}(\alpha)$ (i.e. each r_i odd) of the expression on the RHS is: $(2/\pi)^{3m/2} \cdot (m/r)^{3/2} = 2^{-\Theta(m)} \cdot (m/r)^{3/2}$.

On the other hand, each coefficient of f^t of layer r is equal and the total sum squared of coefficients from layer r of f^t is at most $O(1/r^{3/2})$. Now, using Cauchy Schwarz inequality for the sum of product of fourier coefficients of f^t and g^t at indices corresponding to each valid partition, $r_1, r_2, \ldots, r_m$ of the integer $r > m$ and summing up over all valid partitions of r, the total contribution due to layer r to the correlation is at most: $2^{-\Theta(m)} \cdot 1/r^{3/2}$. Since $\sum_{r>m} 1/r^{3/2}$ converges, we have the claimed upper bound. ◀

◀

As an immediate corollary, we obtain the following hardness for the problem of agnostic learning of halfspaces on the Gaussian distribution.

▶ **Theorem 8** (Hardness of Agnostic Learning of Halfspaces). *Suppose there exists an algorithm $\mathcal{A}$ to learn the class of halfspaces agnostically over the Gaussian distribution to an error of at most ϵ that runs in time $T(n, 1/\epsilon)$. Then, there exists an algorithm to solve k-SLPN that runs in time $\tilde{O}(n \cdot T(n, \frac{2^{O(k)}}{(1-2\eta)})$ where η is the noise rate. In particular, if there is an algorithm that agnostically learns halfspaces on γ_n in time $n^{o(\log(1/\epsilon))}$ then there is an algorithm that solves SLPN for all parities of length $k = O(\log n)$ in time $n^{o(k)}$.*

For a proof, we use Lemma 6 with C as the class of all majorities of length k and note that $\theta(k) = 2^{-\Theta(k)}$.

3.1 Agnostically Learning Sparse Polynomials is Hard

We now reduce k-sparse LPN to agnostically learning degree k and 1-sparse polynomials on the Gaussian distribution and obtain that any algorithm to agnostically learn even a monomial of degree k up to any constant error on the Gaussian distribution runs in time $n^{\Omega(k)}$. We note that the polynomial regression algorithm [22] can be used to agnostically learns degree k polynomials to an accuracy of ϵ in time $n^{O(k)} \cdot \mathsf{poly}(1/\epsilon)$. Thus, our result shows that this running time cannot be improved (assuming that sparse LPN is hard). For a proof, we observe that the Gaussian lift of the parity function χ^γ is noticeably correlated with a sparse polynomial (in fact, just a monomial) on $\mathbb{R}^n$ under the Gaussian distribution. We then invoke Lemma 6 to complete the proof.

▶ **Lemma 9** (Correlation of χ_S^γ with monomials). *Let $M_S : \mathbb{R}^n \to \mathbb{R}$ be the monomial $M_S(x) = \Pi_{i\in S} x_i$. For $\chi_S^\gamma : \mathbb{R}^n \to \{-1, 1\}$, the Gaussian lift of the the parity on $S \subseteq [n]$, we have: $\mathbb{E}_{x\sim\gamma}[\chi_S^\gamma(x) \cdot M_S(x)] = (\frac{2}{\pi})^{|S|/2}$.*

Proof. $\mathbb{E}_{x\sim\gamma}[\chi_S^\gamma(x)\cdot M_S(x)] = (\mathbb{E}_{x_i\sim\gamma}[\mathsf{sign}(x_i)\cdot x_i])^{|}S| = (\mathbb{E}_{z\sim|\gamma|}[z])^{|S|} = (2/\pi)^{|S|/2}$. ◀
Using Lemma 6, we thus have:

▶ **Theorem 10** (Sparse Parity to Sparse Polynomials)**.** *If there is an algorithm to agnostically learn 1-sparse, degree k polynomials on the Gaussian distribution in time $T(n,k,1/\epsilon)$, then, there is an algorithm to solve k-SLPN in time $\tilde{O}(n)\cdot T(n,k,2^{O(k)}/(1-2\eta))$. In particular, if Assumption 1 is true, then any algorithm to agnostically learn degree k monomials up to any constant error runs in time $n^{\Omega(k)}$.*

4 Hardness of Finding Heavy Hermite Coefficients

In this section, we show that a polynomial time algorithm to find all large Hermite coefficients of any function f on the Gaussian distribution using random examples gives a PAC learning algorithm for DNF formulas on the uniform distribution on $\{-1,1\}^n$. The idea is to use the subroutine that recovers large *Hermite coefficients* of a function using random labeled examples to find heavy *Fourier coefficients* of functions on the uniform distribution (via the Gaussian lift) on the hypercube using random examples. Our reduction will then be completed using the properties of the Fourier spectrum of DNF formulas due to [21] (similar to the one used by [13]). Observe that given query access, finding heavy Fourier coefficients on the uniform distribution on $\{-1,1\}^n$ is easy and the reduction yields us a subroutine to find heavy Fourier coefficients by random examples alone.

▶ **Lemma 11.** *Suppose there is an algorithm $\mathcal{A}$, that, for $\epsilon > 0$, uses random examples drawn according to the spherical Gaussian distribution and labeled by an unknown $f : \mathbb{R}^n \to \{-1,1\}$ and returns (with probability at least $2/3$) the Hermite coefficients of f, that are at least ϵ in magnitude in time and samples $T(n,1/\epsilon)$.*

Then, there exists an algorithm, that uses random example access to a Boolean function $g : \{-1,1\}^n \to \{-1,1\}$ on the uniform distribution on $\{-1,1\}^n$, and returns (with probability at least $2/3$), every Fourier coefficient of g of total degree at most d and magnitude at least ϵ, in time and samples $T(n,(2/\pi)^{d/2}/\epsilon)$.

Proof. Given access to random labeled examples from the uniform distribution on $\{-1,1\}^n$ and labeled by a function g, we construct an algorithm $\mathcal{A}'$ which runs $\mathcal{A}$ on a examples labeled by the Gaussian lift, g^γ of g and recovers large Fourier coefficients of g from the set of large Hermite coefficients of g^γ. As before, to simulate a random examples from g^γ we do the following:

1. Draw a random example $(x, g(x))$ where $x \in \{-1,1\}^n$ is uniformly distributed.
2. Draw $y_1, y_2, \cdots, y_n$ as independent half-normals induced by unit variance, zero mean Gaussian.
3. Return $(x \circ y, g(x))$.

Notice that $x \circ y$ is distributed according to the Gaussian distribution. Further,

$$g^\gamma(x) = g(\mathsf{sign}(x_1), \mathsf{sign}(x_2), \cdots, \mathsf{sign}(x_n))$$

for each $x \in \mathbb{R}^n$. Let $\Delta \in \{0,1\}^n \subseteq \mathbb{Z}^n$ (i.e., Δ is an index of a *multilinear* Hermite coefficient). Thus, Δ corresponds to a subset $\beta \subseteq [n]$ such that $|\beta| = d$. We will now show that: $|\widehat{g^\gamma}(\Delta)| \geq (2/\pi)^{-d/2}\epsilon$. We can then run $\mathcal{A}$ to find all Hermite coefficients of g^γ of magnitude at least $(2/\pi)^{-d/2}$, collect all multilinear coefficients of degree at most d and return the corresponding index sets as the indices of the Fourier coefficients of f of magnitude at least ϵ and degree at most d. The Fourier coefficients of g at the indices returned can

then be efficiently computed by taking enough random samples and computing the empirical correlations. This will complete the proof.

For this purpose, note that, being a function on $\{-1,1\}^n$, $g = \sum_{\alpha\subseteq[n]} \hat{g}(\alpha)\cdot\Pi_{i\in\alpha}x_i$. Thus,

$$g^\gamma(x) = g(\mathsf{sign}(x_1),\mathsf{sign}(x_2),\ldots,\mathsf{sign}(x_n)) = \sum_{\alpha\subseteq[n]} \hat{g}(\alpha)\cdot\Pi_{i\in\alpha}\mathsf{sign}(\Pi_{i\in\alpha}x_i).$$

We now have: $\widehat{g^\gamma}_S = \mathbb{E}_{x\sim\gamma_n}[g^\gamma(x)\cdot H_S(x)] = \sum_{T\subseteq[n]}\hat{g}(T)\cdot\mathbb{E}_{x\sim\gamma_n}[\mathsf{sign}(\Pi_{i\in T}x_i)H_S(x)]$. For any $\alpha\neq\beta$ (the subset of $[n]$ corresponding to Δ), then, $\mathbb{E}_{x\sim\gamma_n}[\mathsf{sign}(\Pi_{i\in\alpha}x_i)H_\Delta(x)] = 0$. Thus, using independence of x_i for each $i\in[n]$ and that $\mathbb{E}[|x_i|] = \sqrt{2/\pi}$, we have:

$$\begin{aligned}\mathbb{E}_{x\sim\gamma_n}[g^\gamma(x)\cdot H_\Delta(x)] &= \hat{g}(\beta)\mathbb{E}_{x\sim\gamma_n}[\mathsf{sign}(\Pi_{i\in\beta}x_i)\cdot\Pi_{i\in\beta}x_i]\\ &= \hat{g}(\beta)\Pi_{i\in\beta}\mathbb{E}[|x_i|] = \hat{g}(\beta)(2/\pi)^{-|\beta|/2}.\end{aligned}$$ ◀

We first describe the main idea of the proof: We are given random examples drawn from $\{-1,1\}^n$ and labeled by some function f. We simulate the examples from the Gaussian lift f^γ by embedding the examples from $\{-1,1\}^n$ into $\mathbb{R}^n$ using half-normals as before. We then argue that if $\hat{f}(S)$ is large in magnitude, then so is the multilinear Hermite coefficient at S of f^γ. Thus finding heavy Hermite coefficients of f^γ gives us the indices of large Fourier coefficients of f, which can then be estimated by random sampling. We now provide the details, which are standard and based on [21]. We need the following lemma due to Jackson [21] (we actually state a slightly refined version due to Bshouty and Feldman [7]). In the following, we abuse the notation a little bit and use $\mathcal{D}$ to also refer to the PDF of the distribution denoted by $\mathcal{D}$.

▶ **Lemma 12.** *For any Boolean valued function $f:\{-1,1\}^n\to\{-1,1\}$ computed by a DNF formula of size s, and any distribution $\mathcal{D}$ over $\{-1,1\}^n$, there is $\alpha\subseteq[n]$ such that $|\alpha|\leq\log(2s+1)\cdot||2^n\cdot\mathcal{D}||_\infty$ and $|\hat{f}(\alpha)|\geq\frac{1}{s+1}$.*

On the uniform distribution, the lemma above directly yields a weak learner for DNF formulas. Jackson's key idea here is to observe that learning f on $\mathcal{D}$ is same as learning $2^n f.\mathcal{D}$ on the uniform distribution. Coupled with a boosting algorithm [16, 17, 29] that uses only the distributions for which $||2^n\mathcal{D}||_\infty$ is small ($\mathsf{poly}(1/\epsilon)$), one obtains the PAC learner for DNF formulas.

▶ **Theorem 13.** *If there is an algorithm to find Hermite coefficients of magnitude at least ϵ, of a function $f:\mathbb{R}^n\to\mathbb{R}$ on the Gaussian distribution from random labeled examples in time $\mathsf{poly}(n,1/\epsilon)$, then there is an algorithm to PAC learn DNF formulas on the uniform distribution in polynomial time.*

5 Conclusion and Open Problems

In this paper, we described a general method to embed hard learning problems on the discrete hypercube into the spherical Gaussian distribution on $\mathbb{R}^n$. Using this technique, we showed that any algorithm to agnostically learn the class of halfspaces on the Gaussian distribution runs in time $n^{\Omega(\log(1/\epsilon))}$. We also ruled out a fully polynomial algorithm to agnostically learn sparse polynomials on $\mathbb{R}^n$ complementing the result of Andoni et al. [1] who gave a polynomial time algorithm for learning the class with random additive Gaussian noise.

On the other hand, as described before, the fastest algorithm for agnostically learning halfspaces runs in time $n^{O(1/\epsilon^2)}$ [9]. Thus, an outstanding open problem is to close the gap between these two bounds. That is:

▶ **Question 3.** *What is the optimal time complexity for agnostically learning halfspaces on the Gaussian distribution? In particular, is there an algorithm that agnostically learns halfspaces on the Gaussian distribution in time* $n^{O(\log(1/\epsilon))}$*?*

Acknowledgement. We thank Chengang Wu for numerous discussions during the preliminary stages of this work. We thank the anonymous reviewers for pointing out the typos in a previous version of this paper.

References

1 Alexandr Andoni, Rina Panigrahy, Gregory Valiant, and Li Zhang. Learning sparse polynomial functions. In *SODA*, 2014.
2 Shai Ben-David and Hans-Ulrich Simon. Efficient learning of linear perceptrons. In *NIPS*, pages 189–195, 2000.
3 Quentin Berthet and Philippe Rigollet. Complexity theoretic lower bounds for sparse principal component detection. In *COLT*, pages 1046–1066, 2013.
4 Aharon Birnbaum and Shai Shalev-Shwartz. Learning halfspaces with the zero-one loss: Time-accuracy tradeoffs. In *NIPS*, pages 935–943, 2012.
5 A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM*, 50(4):506–519, 2003.
6 Avrim Blum, Alan M. Frieze, Ravi Kannan, and Santosh Vempala. A polynomial-time algorithm for learning noisy linear threshold functions. *Algorithmica*, 22(1/2):35–52, 1998.
7 Nader H. Bshouty and Vitaly Feldman. On using extended statistical queries to avoid membership queries. *Journal of Machine Learning Research*, 2:359–395, 2002.
8 Amit Daniely, Nati Linial, and Shai Shalev-Shwartz. From average case complexity to improper learning complexity. *CoRR*, abs/1311.2272, 2013.
9 Ilias Diakonikolas, Daniel M. Kane, and Jelani Nelson. Bounded independence fools degree-2 threshold functions. *CoRR*, abs/0911.3389, 2009.
10 Ilias Diakonikolas, Ryan O'Donnell, Rocco A. Servedio, and Yi Wu. Hardness results for agnostically learning low-degree polynomial threshold functions. In *SODA*, pages 1590–1606, 2011.
11 Ilias Diakonikolas, Ryan O'Donnell, Rocco A. Servedio, and Yi Wu. Hardness results for agnostically learning low-degree polynomial threshold functions. In *SODA*, pages 1590–1606, 2011.
12 V. Feldman. A complete characterization of statistical query learning with applications to evolvability. *Journal of Computer System Sciences*, 78(5):1444–1459, 2012.
13 V. Feldman, P. Gopalan, S. Khot, and A. Ponuswami. On agnostic learning of parities, monomials and halfspaces. *SIAM Journal on Computing*, 39(2):606–645, 2009.
14 Vitaly Feldman and Varun Kanade. Computational bounds on statistical query learning. In *COLT*, pages 16.1–16.22, 2012.
15 Vitaly Feldman, Pravesh Kothari, and Jan Vondrák. Representation, approximation and learning of submodular functions using low-rank decision trees. In *COLT*, pages 711–740, 2013.
16 Yoav Freund. Boosting a weak learning algorithm by majority. In *COLT*, pages 202–216, 1990.
17 Yoav Freund. An improved boosting algorithm and its implications on learning complexity. In *COLT*, pages 391–398, 1992.
18 Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

19 Venkatesan Guruswami and Prasad Raghavendra. Hardness of learning halfspaces with noise. *SIAM J. Comput.*, 39(2):742–765, 2009.

20 D. Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.

21 Jeffrey C. Jackson. An efficient membership-query algorithm for learning dnf with respect to the uniform distribution. *J. Comput. Syst. Sci.*, 55(3):414–440, 1997.

22 Adam Tauman Kalai, Adam R. Klivans, Yishay Mansour, and Rocco A. Servedio. Agnostically learning halfspaces. *SIAM J. Comput.*, 37(6):1777–1805, 2008.

23 Daniel M. Kane, Adam Klivans, and Raghu Meka. Learning halfspaces under log-concave densities: Polynomial approximations and moment matching. In *COLT*, pages 522–545, 2013.

24 M. Kearns, R. Schapire, and L. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

25 Eike Kiltz, Krzysztof Pietrzak, David Cash, Abhishek Jain, and Daniele Venturi. Efficient authentication from hard learning problems. In *EUROCRYPT*, pages 7–26, 2011.

26 Adam Klivans, Pravesh Kothari, and Igor Oliveira. Constructing hard functions from learning algorithms. *Conference on Computational Complexity, CCC*, 20:129, 2013.

27 Adam R. Klivans, Ryan O'Donnell, and Rocco A. Servedio. Learning intersections and thresholds of halfspaces. *J. Comput. Syst. Sci.*, 68(4):808–840, 2004.

28 Adam R. Klivans, Ryan O'Donnell, and Rocco A. Servedio. Learning geometric concepts via gaussian surface area. In *FOCS*, pages 541–550, 2008.

29 Adam R. Klivans and Rocco A. Servedio. Boosting and hard-core set construction. *Machine Learning*, 51(3):217–238, 2003.

30 Adam R. Klivans and Alexander A. Sherstov. Cryptographic hardness for learning intersections of halfspaces. In *FOCS*, pages 553–562, 2006.

31 Adam R. Klivans and Alexander A. Sherstov. Lower bounds for agnostic learning via approximate rank. *Computational Complexity*, 19(4):581–604, 2010.

32 Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM J. Comput.*, 22(6):1331–1348, 1993.

33 Ryan O'Donnell. Fourier coefficients of majority. `http://www.contrib.andrew.cmu.edu/~ryanod/?p=877`, 2012.

34 Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

35 Shai Shalev-Shwartz, Ohad Shamir, and Karthik Sridharan. Learning kernel-based halfspaces with the zero-one loss. In *COLT*, pages 441–450, 2010.

36 Gregory Valiant. Finding correlations in subquadratic time, with applications to learning parities and juntas. In *The 53rd Annual IEEE Symposium on the Foundations of Computer Science (FOCS)*, 2012.

37 L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

38 V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, New York, 1998.

A Finding Large Hermite Coefficients Using Queries

For $\Delta_1 \in \mathbb{Z}^k$ and $\Delta_2 \in \mathbb{Z}^{n-k}$, let $\Delta = \Delta_1 \circ \Delta_2$ denote the n-tuple obtained by concatenating Δ_1 and Δ_2. Similarly, for $s \in \mathbb{R}^k$ and $z \in \mathbb{R}^{n-k}$ let $t = s \circ z$ denote the element of $\mathbb{R}^n$ obtained by concatenating s and z. We are now ready to present the procedure to find heavy Hermite coefficients of a function given query access to it. Since heavy Hermite coefficients, in general, may not be multilinear, we adapt the idea of [32] to work in this setting. Our proof is based on that of [32] (see also the lecture notes by O'Donnell [33]).

▶ **Theorem 14.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be any function such that $||f||_2 = \mathbb{E}_{x\sim\gamma}[f(x)^2] = 1$. There exists an algorithm that uses query access to f and runs in time $\tilde{O}(nd/\epsilon^2)$ to return every index $\Delta \in \mathbb{Z}^k$ of degree d such that $|\hat{f}(\Delta)| \geq \epsilon$.*

Proof. We estimate every coefficient of f that is larger than ϵ within an error of $\epsilon/3$. Thus, for each $\Delta \in \mathbb{Z}^n$, we will obtain $\tilde{f}(\Delta)$ such that $|\tilde{f}(\Delta) - \hat{f}(\Delta)| \leq \epsilon/3$.

We first describe a subroutine which we will repeatedly use in the algorithm. For any $\Delta \in Z^k$, let

$$W_{\Delta_1} = \sum_{\Delta_2 \in \mathbb{Z}^{n-k}} \hat{f}(\Delta_1 \circ \Delta_2)^2.$$

▶ **Lemma 15.** *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a function with query access. Given $\Delta_1 = \{i_1, i_2, \cdots, i_k\} \in \mathbb{Z}^k$ such that $\sum_{j\leq k} i_j \leq d$, there is an algorithm that returns a value v such that $|v - \sum_{T:T\in\mathbb{N}^{n-k}} \hat{f}(\Delta_1 \circ \Delta_2)^2| \leq \delta$ with probability at least $2/3$ in time and queries $\tilde{O}(\frac{nd}{\delta^2})$.*

Proof. Define $\hat{f}_{\Delta_1} : \mathbb{R}^{n-k} \to \mathbb{R}$ by

$$\hat{f}_{\Delta_1}(z) = \mathbb{E}_{x\sim\mathbb{R}^k}[f(x \circ z) \cdot H_{\Delta_1}(x)]. \tag{3}$$

For $W \in \mathbb{Z}^n$, let $W_k \in \mathbb{R}^k$ denote the first k coordinate values of W and W_{n-k} denote the last k. One then has $H_W(x \circ z) = H_{W_k}(x) \cdot W_{W_{n-k}}(z)$ for any $x \in \mathbb{R}^{W_k}$ and $z \in \mathbb{R}^{W_{n-k}}$. Then, we have:

$$\begin{aligned}
\hat{f}_{\Delta_1}(z) &= \mathbb{E}_{x\sim\mathbb{R}^k}[f(x \circ z) \cdot H_{\Delta_1}(x)] \\
&= \mathbb{E}_{x\sim\mathbb{R}^k}[\sum_{W\in\mathbb{Z}^n} \hat{f}(W) \cdot H_W(x \circ z) \cdot H_{\Delta_1}(x)] \\
&= \mathbb{E}_{x\sim\mathbb{R}^k}[\sum_{W\in\mathbb{Z}^n} \hat{f}(W) \cdot H_{W_k}(x) H_{W_{n-k}}(z) \cdot H_{\Delta_1}(x)] \\
&= \sum_{W\in\mathbb{Z}^n} \mathbb{E}_{x\sim\mathbb{R}^k}[\hat{f}(W) \cdot H_{W_k}(x) H_{W_{n-k}}(z) \cdot H_{\Delta_1}(x)]
\end{aligned}$$

For every W such that $W_k \neq S$, the term above evaluates to 0 due to the orthogonality of H_{Δ_1} and H_{W_k}

$$= \sum_{W=\Delta_1 \; circ \Delta_2} \hat{f}(\Delta_1 \circ \Delta_2) H_{\Delta_2}(z) \tag{4}$$

Now,

$$\begin{aligned}
\sum_{\Delta_2\in\mathbb{N}^{n-k}} \hat{f}(\Delta_1 \circ \Delta_2)^2 &= \mathbb{E}_{z\in\gamma^{n-k}}[(\sum_{\Delta_2\in\mathbb{N}^{n-k}} \hat{f}(\Delta_1 \circ \Delta_2) H_{\Delta_2}(z))^2] \\
&= \mathbb{E}_{z,z'\in\gamma^{n-k}}[(\sum_{\Delta_2\in\mathbb{N}^{n-k}} \hat{f}(\Delta_1 \circ \Delta_2) H_{\Delta_2}(z)) \cdot \\
&\qquad (\sum_{\Delta_2\in\mathbb{N}^{n-k}} \hat{f}(\Delta_1 \circ \Delta_2) H_{\Delta_2}(z'))] \\
&\text{Using Equation (4)} \\
&= E_{z,z'\in\gamma^{n-k}}[\hat{f}_{\Delta_2}(z) \cdot \hat{f}_{\Delta_2}(z')] \\
&\text{Using Equation (3)} \qquad (5) \\
&= \mathbb{E}_{x,x'\in\gamma^k,\; z,z'\in\gamma^{n-k}}[f(x \circ z) f(x' \circ z') H_{\Delta_1}(x) \cdot H_{\Delta_1}(x')] \qquad (6)
\end{aligned}$$

The quantity in the RHS of Equation (6) can be computed up to an additive error of at most $\delta > 0$ by drawing $\tilde{O}(1/\delta^2)$ random points from $\mathbb{R}^k$ and $\mathbb{R}^{n-k}$ and obtaining the values of f at the appropriate combinations using queries. Thus, we obtain the required result. ◀

We can now describe the algorithm:

1. Set $\delta = \epsilon^2/3$, $\beta = \frac{1}{3dn\epsilon^2}$.
2. $\mathbb{S} \leftarrow \emptyset$.
3. For $j = 1$ to n
   a. For $k = 1$ to d:
      i. For each $T \in \mathbb{S}$, if $Z = T \circ k$ is such that H_Z is of degree at most d:
         A. Estimate W_Z to an accuracy of δ to a confidence of $1 - \beta$.
         B. If $W_Z > \epsilon^2/2$, $\mathbb{S} \leftarrow \mathbb{S} \cup Z$.
4. Return $\mathbb{S}$.

The algorithm above is analogous to the Kushilevitz Mansour algorithm and it is easy the correctness based on the lemma above: We begin by noting the sum squared of all Hermite coefficients of f is 1 as the Hermite transformation preserves ℓ_2 norms. Thus, the number of coefficients that are larger than ϵ in magnitude are at most $1/\epsilon^2$. One can thus argue that with high probability, the size of $\mathbb{S}$ in the algorithm above is at most $O(1/\epsilon^2)$ at all times. In the j^{th} iteration, the algorithm tries to append any of the d powers of x_j to each of the indices in $\mathbb{S}$. For each such newly produced index Z, the algorithm estimates the Weight W_Z as in the lemma above. It adds Z to $\mathbb{S}$ whenever W_Z is estimated to be higher than $\epsilon^2/2$. Thus, each such iteration needs $O(nd)$ time to execute.

This completes the proof. ◀

Smoothed Analysis on Connected Graphs*

Michael Krivelevich[1], Daniel Reichman[2], and Wojciech Samotij[3]

1 **School of Mathematical Sciences**
Tel Aviv University, Tel Aviv 69978, Israel
krivelev@post.tau.ac.il

2 **Department of Computer Science and Applied Mathematics**
Weizmann Institute of Science, Rehovot, Israel
daniel.reichman@gmail.com

3 **School of Mathematical Sciences and Trinity College**
Tel Aviv University Tel Aviv 69978, Israel; and Cambridge CB2 1TQ, UK
samotij@post.tau.ac.il

Abstract

The main paradigm of smoothed analysis on graphs suggests that for any large graph G in a certain class of graphs, perturbing slightly the edges of G at random (usually adding few random edges to G) typically results in a graph having much "nicer" properties. In this work we study smoothed analysis on trees or, equivalently, on connected graphs. Given an n-vertex connected graph G, form a random supergraph G^* of G by turning every pair of vertices of G into an edge with probability $\frac{\varepsilon}{n}$, where ε is a small positive constant. This perturbation model has been studied previously in several contexts, including smoothed analysis, small world networks, and combinatorics.

Connected graphs can be bad expanders, can have very large diameter, and possibly contain no long paths. In contrast, we show that if G is an n-vertex connected graph then typically G^* has edge expansion $\Omega(\frac{1}{\log n})$, diameter $O(\log n)$, vertex expansion $\Omega(\frac{1}{\log n})$, and contains a path of length $\Omega(n)$, where for the last two properties we additionally assume that G has bounded maximum degree. Moreover, we show that if G has bounded degeneracy, then typically the mixing time of the lazy random walk on G^* is $O(\log^2 n)$. All these results are asymptotically tight.

1998 ACM Subject Classification Randomness, geometry and discrete structures

Keywords and phrases Random walks and Markov chains, Random network models

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.810

1 Introduction

In this paper, we consider the following model of randomly generated graphs. We are given a fixed undirected graph $G = (V, E)$ on n vertices. For every pair $f \in \binom{V}{2}$, we add f to G, independently of all other pairs, with probability $\frac{\varepsilon}{n}$, where ε is a small (yet fixed) positive constant. Let R be the set of edges added and consider the random graph

$$G^* := (V, E \cup R).$$

* This work was partially supported by the USA-Israel BSF Grant 2010115, by grant 912/12 from the Israel Science Foundation, by the Israel Science Foundation (grant No. 621/12), by the I-CORE Program of the Planning and Budgeting Committee and The Israel Science Foundation (grant No. 4/11), by the ERC Advanced Grant DMMCA and grants from the Israel Science Foundation.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 810–825

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

This model can be viewed as a generalization of the classical Erdős-Rényi random graph, where one starts from an empty graph and adds edges between all possible pairs of vertices independently with a given probability. The focus on "small" ε means that we are interested in the effect of a rather gentle random perturbation. In particular, the average degree of G^* is (typically) close to that of G (assuming that G is connected, for example). Studying the effect of small perturbations on graphs, matrices, and other structures arises in diverse settings in several fields such as combinatorics, design and analysis of algorithms, linear algebra, and mathematical programming. We refer the reader to Section 1.3 for more details.

In this work, we study several properties of G^*, when G is *connected*. We first need a few definitions. For a graph $G = (V, E)$ and a subset $S \subseteq V$, we denote by ∂S the set of all edges of G with exactly one endpoint in S. We define $N(S)$ to be the set of all vertices in $V \setminus S$ that have a neighbor in S. When the graph G is not clear from the context, we will use the notation $\partial_G S$ and $N_G(S)$ to avoid ambiguity. The *edge*-isoperimetric number of G (also known as the *Cheeger constant*), denoted $c(G)$, is defined by

$$c(G) := \min\left\{\frac{|\partial(U)|}{|U|} : 0 < |U| \leqslant \frac{|V|}{2}\right\}.$$

Similarly, the *vertex*-isoperimetric number of G, denoted $\iota(G)$, is defined by

$$\iota(G) := \min\left\{\frac{|N(U)|}{|U|} : 0 < |U| \leqslant \frac{|V|}{2}\right\}.$$

Somewhat informally we shall refer to $c(G)$ and $\iota(G)$ as the edge and the vertex expansions of G, respectively. Observe that $\iota(G) \geqslant c(G)/\Delta(G)$ where $\Delta(G)$ is the maximum degree of G. Hence, when $\Delta(G)$ is bounded by a constant, then the vertex and edge expansions of G have the same order of magnitude. On the other hand, there are n-vertex graphs G for which $\iota(G) = O(c(G)/n)$. The Cheeger constant has been studied extensively as it is related to a host of combinatorial properties of the underlying graph. In particular, there is a strong connection between the Cheeger constant of G and the mixing time of the lazy random walk on G.

1.1 Our Results

We begin by describing our results regarding the expansion properties of perturbed connected graphs.

For every *connected* n-vertex graph G, it holds that $\iota(G) = \Omega(\frac{1}{n})$ as every subset $S \subseteq V$ has at least one neighbor outside S. Moreover, if G is a tree, then $\iota(G) = O(\frac{1}{n})$. Our first result is that for every *connected* graph with bounded maximum degree, the random perturbation G^* asymptotically almost surely[1] (a.a.s.) satisfies $\iota(G^*) = \Omega(\frac{1}{\log n})$.

▶ **Theorem 1.** *For every $\varepsilon > 0$, there exists $\delta > 0$ such that the following holds. Let G be an n-vertex connected graph with maximum degree Δ. If $R \sim G(n, \frac{\varepsilon}{n})$, then a.a.s. the graph $G^* = G \cup R$ has vertex expansion at least $\frac{\delta}{\Delta^3 \log n}$.*

We note that in general one cannot remove restrictions on the maximum degree entirely. To see this, consider the case when $G = K_{1,n-1}$. After adding to G *any* εn edges, there will be an independent set S with at least $(1 - 2\varepsilon)n$ vertices such that $|N(S)| = 1$.

We obtain a similar bound on the *edge*-expansion without any assumptions on the maximum degree.

[1] That is, with probability tending to 1 as the number of vertices n tends to infinity.

▶ **Theorem 2.** *For every $\varepsilon > 0$ and $\alpha < 1$, there exists $\delta > 0$ such that the following holds. Let G be an n-vertex connected graph, choose $R \sim G(n, \frac{\varepsilon}{n})$, and let $G^* = G \cup R$. Then a.a.s. for every set $S \subseteq V(G)$ with $|S| \leqslant \alpha n$,*

$$|\partial_{G^*} S| \geqslant \frac{\delta}{\log(en/|S|)}|S|.$$

In particular, $c(G^*) \geqslant \frac{\delta}{\log(en)}$.

It should be noted that Theorem 2 implies that a.a.s. the vertex expansion of G^* is at least $\frac{\delta}{\Delta(G)\log(en)}$. This improves the bound obtained in Theorem 1 when $\Delta(G) \gg \left(\frac{\log n}{\log\log n}\right)^{1/3}$; to see this, observe that $G(n, \epsilon/n)$ a.a.s. contains vertices of degree $\Omega\left(\frac{\log n}{\log\log n}\right)$.

Furthermore, we prove an even stronger bound on the edge expansion of connected subsets of a perturbed connected graph.

▶ **Theorem 3.** *For every $\varepsilon > 0$ and $\alpha < 1$, there exist $\delta > 0$ and $K > 0$ such that the following holds. Let G be an n-vertex connected graph, choose $R \sim G(n, \varepsilon/n)$, and let $G^* = G \cup R$. Then a.a.s. for every connected (in G^*) set $S \subseteq V(G)$ with $K \log n \leqslant |S| \leqslant \alpha n$,*

$$|\partial_{G^*} S| \geqslant \delta|S|.$$

We consider sets of size at most αn for an arbitrary $\alpha < 1$ (instead of restricting our attention to sets of size at most $n/2$, as is customary in dealing with edge expansion) since this allows us later to give upper bounds on the conductance of sets of volume up to a half of the total volume which is crucial for the proof of Theorem 5 stated below. Here volume is measured in terms of the degree sum rather than the number of vertices.

Using Theorem 3, we derive the following upper bound on the diameter of a randomly perturbed connected graph. Observe that the diameter of a (non-perturbed) n-vertex connected graph may be as high as $n - 1$ (when the graph is a path on n vertices).

▶ **Theorem 4.** *For every $\varepsilon > 0$, there exists $C > 0$ such that the following holds. Let G be an n-vertex connected graph, choose $R \sim G(n, \frac{\varepsilon}{n})$, and let $G^* = G \cup R$. Then a.a.s. the diameter of G^* is at most $C \log n$.*

Flaxman and Frieze [14] proved an upper bound of $O(\log n)$ on the diameter of randomly perturbed strongly connected *digraphs*. The result of [14] requires that the maximum degree of the base graph is upper bounded by some function of n (see Section 1.3 for details). Unlike their work, our upper bound on the diameter of G^* holds unconditionally, regardless of the maximum degree of the base graph G.

Using Theorem 3, we also prove upper bounds on the mixing times of lazy random walks on randomly perturbed connected graphs. Recall the notion of *degeneracy*. Given a positive integer D, a graph G is called *D-degenerate* if every subgraph of G contains a vertex of degree at most D. Observe that every graph G is $\Delta(G)$-degenerate and trees are 1-degenerate. Also, if G is D-degenerate, then every subset $S \subseteq V(G)$ spans at most $D|S|$ edges. Using the machinery developed by Fountoulakis and Reed [16], we are able to prove the following bound on the mixing time of the lazy random walk on a random perturbation of a connected graph with bounded degeneracy.

▶ **Theorem 5.** *For all positive D and ε, there exists a constant M such that the following holds. Let G be an n-vertex D-degenerate connected graph, choose $R \sim G(n, \frac{\varepsilon}{n})$ and let $G^* = G \cup R$. Then a.a.s.*

$$T_{\mathrm{mix}}(G^*) \leqslant M \log^2 n.$$

For a precise definition of T_{mix}, we refer the reader to Section 2. The bound in Theorem 5 above is tight when G is the path on n vertices, as then a.a.s. G^* contains an induced subgraph which is a path of length $\Omega(\log n)$. Moreover, we cannot expect that $T_{\mathrm{mix}}(G^*) = O(\log^2 n)$ for an arbitrary connected graph G, as the following example demonstrates. Let G be the graph obtained by connecting two disjoint cliques of order $n/2$ with a single edge and let $R \sim G(n, \frac{1}{n})$. As the number of edges interconnecting the two cliques in the perturbed graph is a.a.s. $O(n)$, the conductance of G^* is $O(\frac{1}{n})$, which implies via standard results (e. g., [22]) that the mixing time of the lazy random walk on G^* is $\Omega(n)$.

The effect of small random perturbations on connected graphs from several families has been studied before, see, e. g., [1, 24]. In particular, a $O(\log^2 n)$ bound (holding a.a.s.) on the mixing time of a simple random walk on a random perturbation of the ring graph was proved in [1], see Section 1.3 for more details. Our Theorem 5 demonstrates that an upper bound of $O(\log^2 n)$ on the mixing time (holding a.a.s.) is a rather general phenomenon for perturbed connected graphs.

Finally, we establish the existence of long paths in perturbed connected graphs with bounded maximum degree. Observe that a connected bounded degree graph with n vertices might contain only paths of length $O(\log n)$, as the case of the complete binary tree demonstrates.

▶ **Theorem 6.** *For every $\varepsilon, \Delta > 0$, there exists $c > 0$ such that the following holds. Let G be an n-vertex connected graph with maximum degree bounded by Δ. Form a random graph $R \sim G(n, \frac{\varepsilon}{n})$, and let $G^* = G \cup R$. Then G^* a.a.s. contains a path of length cn.*

The assumption that the maximum degree is bounded is crucial, as it is easy to see that if $G = K_{1,n-1}$ and $\varepsilon < 1$, then a.a.s. the length of a longest path in $G \cup R$ is $O(\log n)$. This follows as it is known that a.a.s. each connected component of $G(n, \frac{\varepsilon}{n})$ has $O(\log n)$ vertices and the vertex set of any simple path in G^* intersects at most two connected components in R.

Finally, one may ask what happens if one incorporates edge deletions in our model. Consider the case when G is an n-vertex tree with $\Omega(n)$ leaves, e. g., G is a complete binary tree over n vertices. If we now add and remove edges randomly with probability $\frac{\varepsilon}{n}$, then with constant probability, we will isolate one of the leaves of G (as the probability of isolating a fixed vertex with degree one in G is about $\frac{\varepsilon}{n} \cdot e^{-\varepsilon}$). Hence we cannot expect the graph resulting after perturbations in this case to have nontrivial expansion properties.

1.2 Our Techniques

In proving Theorem 1, we use a fairly basic result (see e. g., [19]) to decompose graph of bounded degree to disjoint connected sets of comparable sizes. Treating each of these sets as a 'super-vertex' allows us to view the auxiliary graph induced by the random edges between sets as essentially the standard binomial random graph whose edge probability should be now compared to the number of super-vertices as opposed to the (much larger) number of vertices. Consequently, standard methods and results regarding the threshold for connectivity and the existence of long paths in binomial random graphs can be used.

In order to deal with the Cheeger constant of perturbed graphs, we prove a new upper bound on the number of connected subsets of given cardinality and number of vertices in their boundary. We believe that this bound (stated below), which we prove using an elementary argument, may be of independent interest. Recall that a subset of vertices of a graph is *connected*, if it induces a connected subgraph.

▶ **Proposition 7.** *Let G be an arbitrary graph and let $v \in V(G)$. For integers a and b, let $\mathcal{C}(v, a, b)$ denote the collection of connected subsets A of $V(G)$ such that $v \in A$, $|A| = a$, and $|N(A)| = b$. Then*

$$|\mathcal{C}(v, a, b)| \leqslant \binom{a+b-1}{b}.$$

We remark that the bound in Proposition 7 is tight for all values of a and b. To see this, consider the case when $G = K_{1,a+b-1}$ and v is the center vertex.

In bounding the mixing time, we rely on an upper bound on the mixing time of a lazy random walk due to Fountoulakis and Reed [16]. This bound, which they used [17] to upper bound the mixing time of the lazy random walk on the giant component of $G(n, p)$, is suited for bounding the mixing time of random walks on graphs whose large vertex sets expand well but small sets (e. g., of logarithmic size) do not have to. Another attractive feature of the result of Fountoulakis and Reed is that it allows one to focus on the conductance of connected sets, which significantly simplifies union bound estimates. We note that the classical work of Jerrum and Sinclair [18] for upper-bounding the mixing time T_{mix} in terms of the conductance Φ of G (see Section 2 for precise definitions), namely

$$T_{\text{mix}} \leqslant O\left(\frac{\log n}{\Phi^2}\right)$$

would give in our setting a weaker bound of $O(\log^3 n)$.

1.3 Related Work

The study of random perturbations of graphs arose in several contexts. One of them is the field of *smoothed analysis*, which originated from the work of Spielman and Teng [28] on the smoothed complexity of the simplex algorithm. This field attempts to provide a theoretical explanation for the good performance of certain heuristics on "real-life" instances based on the assumption that they are likely to be subjected to random perturbations. It has been applied to a host of other problems such as numerical analysis and linear algebra [26, 30], machine learning [5], and satisfiability [9, 12]. It is closely related to the study of random perturbations of combinatorial structures and devising efficient algorithms for such "semi-random" instances, which had been considered in the past, see [6, 7, 13, 15, 21, 27, 29].

Another context where the study of random perturbations naturally arose, is the field of small world networks, see [11, 24, 25]. In an attempt to model social networks arising in "real-life" settings, one studies properties of networks composed of a (usually sparse) connected "base" graph along with a set of random edges, where every random edge is added independently with probability p. One well-known example is the Newmann–Watts small world model [24, 25] (NW small world for short), where the base graph is the (n, k)-ring, i. e., the graph with vertex set $\{0, \ldots, n-1\}$ and edge set $\big\{\{i, j\} : i + 1 \leqslant j \leqslant i + k\big\}$ (where addition is modulo n) and p is equal to $\frac{c}{n}$ for some constant $c > 0$.

Durrett [11] showed that with high probability the mixing time of the lazy random walk on the NW small world is upper-bounded by $O(\log^3 n)$ and lower-bounded by $\Omega(\log^2 n)$. These results were improved by Addario-Berry and Lei [1] who proved that this mixing time is a.a.s. $O(\log^2 n)$. It is worth noting that our approach is similar to [1] in the sense that we bound the conductance of connected sets and then use this upper bound with the results of [16] to bound the mixing time. The crucial difference between our proof and theirs is the technique of counting connected sets with small boundary. While [1] uses a somewhat

involved argument based on the Lagrange inversion formula, we use a more elementary approach based on Proposition 7.

Similar ideas were used in the study of the mixing time of the simple random walk on the giant component in a supercritical random graph $G(n, \frac{1+\varepsilon}{n})$. Fountolakis and Reed [16] and Benjamini, Kozma, and Wormald [3] showed that a.a.s. this mixing time is $O(\log^2 n)$. Moreover, there has been interest in probability theory in studying the robustness of the mixing time under random perturbations, see [4, 10].

Flaxman [15] examined the edge expansion of several models of randomly perturbed graphs. In particular, he considered the model studied in this work. He showed in particular that if $G = (V, E)$ is an n-vertex connected graph and $R \sim G(n, \frac{\varepsilon}{n})$, then a.a.s. all linear sized vertex subsets $S \subseteq V$, $|S| \leqslant n/2$, send outside at least a linear in n number of edges in $G^* = G \cup R$. The effect of adding random edges on the diameter of a given graph was considered by Bollobás and Chung [8], who proved that adding a random matching to an n-vertex cycle result a.a.s. with a graph with diameter $(1 + o(1)) \log_2 n$. The case of *directed* graphs was considered by Flaxman and Frieze [14]. They proved that if D is an n-vertex strongly connected digraph with maximum degree bounded by $n^{\frac{\varepsilon}{100}}$ and $R \sim D(n, \frac{\varepsilon}{n})$, then a.a.s. the diameter of $D \cup R$ is at most $100\varepsilon^{-1} \log n$. Our proof idea is different from theirs.

1.4 Outline of the Paper

In Section 2, we fix some notation, give a precise definition of the mixing time of a random walk, and state two auxiliary probabilistic lemmas that are used later in the paper. In Sections 3, 4 and 5, we prove Theorems 2, 4 and 5, respectively. Section 3 contains also the a proof of Proposition 7. The proofs of Theorems 1, 3 and 6 can be found in the Appendix. In Section 6, we state several concluding remarks.

2 Preliminaries

Let G be a graph with vertex set V. Given two disjoint sets $A, B \subseteq V$, we denote by $E(A, B)$ the set of all edges with one endpoint in A and one endpoint in B and by $E(A)$ the set of all edges entirely contained in A. We will denote the cardinality of $E(A)$ by $e(A)$. The degree of a vertex v in G is denoted by $\deg(v)$ and the maximum degree of G is denoted by $\Delta(G)$.

We denote by $[n]$ the set $\{1, \dots, n\}$. When dealing with an n-vertex graph, we will implicitly assume that that its vertex set is $[n]$. We denote by $G(n, p)$ the classical binomial random graph with vertex set $[n]$ and edge probability p. Given a graph property $\mathcal{P}$ and a sequence (μ_n), where μ_n is a probability distribution over n-vertex graphs, we will say that $\mathcal{P}$ holds asymptotically almost surely (a.a.s.) if $\lim_{n\to\infty} \Pr_{G\sim\mu_n}(G \in \mathcal{P}) = 1$.

The lazy random walk on a graph $G = (V, E)$ is the Markov chain defined as follows. The set of states is V. For any vertex $u \in V$, the walk stays in u with probability $\frac{1}{2}$ and with probability $\frac{1}{2}$, it moves to a uniformly chosen random neighbor v of u (so that the transition probability $\Pr(u \to v)$ is $\frac{1}{2\deg(u)}$). When G is connected, this Markov chain is well-known to be irreducible and ergodic and hence it converges to a stationary distribution π which can be seen to equal $\pi(u) = \frac{\deg(u)}{2|E|}$ for every $u \in V$, see [22]. We will be interested in estimating how quickly this random walk on G converges to its stationary distribution π. To this end, we recall that the *total variation distance* d_{TV} between two distributions p_1, p_2 on V is defined by

$$d_{\mathrm{TV}}(p_1, p_2) := \max_{A\subseteq V} |p_1(A) - p_2(A)|.$$

Let P be the transition matrix of the random walk. The mixing time $T_{\mathrm{mix}}(G)$ is defined by

$$T_{\mathrm{mix}}(G) := \sup_{x_0} \min \left\{ t \colon d_{TV}(x_0 P^t, \pi) \leqslant \frac{1}{4} \right\},$$

where the supremum is taken over all probability distributions x_0 on V.

The proof of the following Lemma can be found in the Appendix.

▶ **Lemma 8.** *For every $C \geqslant 1$, if $p \leqslant C/n$, then a.a.s. for every non-empty set S of vertices in $G(n,p)$, we have $e(S) < 2C|S|$. In particular, if $p \leqslant \frac{1}{n}$, then a.a.s. for every non-empty set S of vertices in $G(n,p)$, we have $e(S) < 2|S|$.*

We close this section with a version of Chernoff's inequality (see, e. g., [23]).

▶ **Lemma 9.** *Suppose that $X = \sum_{i=1}^{m} X_i$ where every X_i is a $\{0,1\}$-random variable with $\Pr(X_i = 1) = p$ and the X_is are jointly independent. Then for arbitrary $\eta \in (0,1)$,*

$$\Pr(X < (1-\eta)pm) \leqslant \exp(-pm\eta^2/2).$$

3 Edge Expansion

In this section, we prove Proposition 7 and derive from it Theorem 2.

Proof of Proposition 7. Assume that the vertices of G are labeled with distinct integers. We will describe an algorithm that, given an $A \in \mathcal{C}(v,a,b)$, outputs an encoding of A using a sequence of $a-1$ ones and b zeros in such a way that no two sets are encoded with the same sequence. This will clearly imply the statement of the proposition.

Let $S = \{v\}$ and $B = \emptyset$. The algorithm will grow the sets S and B, adding one vertex to one of the sets in each of its $a+b-1$ iterations, making sure that the invariants $S \subseteq A$ and $B \subseteq N(A)$ hold in every iteration. It will stop when $S = A$ and $B = N(A)$, after having moved $a-1$ vertices to S and b vertices to B. For the sake of brevity, we will denote by T the set $N(S) \setminus B$, updated after each iteration. Intuitively, in every iteration, S is the set of vertices that are known to belong to A, B is the set of vertices that belong to $N(A)$, and T is the remaining set of vertices for which we do not know yet whether they belong to A or to $N(A)$.

While $S \neq A$ or $B \neq N(A)$, we repeat the following. Let w be the vertex with the smallest label in T. Note that the assumption that A is connected implies that T is non-empty. Consider two cases. If $w \in A$, then move w to S and append 1 to the sequence encoding A. Otherwise, if $w \notin A$, then move w to B and append 0 to the sequence encoding A. Note that in this case $w \in N(A)$, since $S \subseteq A$ and $w \in N(S) \setminus A$.

A moment of thought reveals that decoding can be performed in an analogous way and given v and the $\{0,1\}$-sequence encoding A, one can recover the set A. This completes the proof. ◀

Proof of Theorem 2. For positive integers s, m, and b, denote by $\mathcal{S}(s,m,b)$ the collection of all sets S of s vertices such that in the graph G, the set S induces exactly m connected components and the sum of their vertex boundaries is exactly b. In other words, $\mathcal{S}(s,m,b)$ consists of all sets $S \subseteq V(G)$ such that there is a partition $S = S_1 \cup \ldots \cup S_m$, where each S_i is connected, there are no edges of G connecting different S_i, and $|N_G(S_1)| + \ldots + |N_G(S_m)| = b$. Since G is connected, each $S \in \mathcal{S}(s,m,b)$ satisfies $|\partial_G S| \geqslant b \geqslant m \geqslant 1$ (but not necessarily

$|N_G(S)| \geqslant b$). Therefore, it is enough to show that there exist positive constants K and δ such that a.a.s. for every s satisfying $s \geqslant K \log n$,

$$|\partial_R S| \geqslant \frac{\delta s}{\log(en/s)} \quad \text{for all } S \in \mathcal{S}(s,m,b) \text{ with } m \leqslant b \leqslant \frac{\delta s}{\log(en/s)}. \tag{1}$$

(For small sets S with $|S| \leqslant K \log n$, we have that $|\partial_{G^*} S| \geqslant |\partial_G S| \geqslant 1 \geqslant \frac{\delta s}{\log(en/s)}$, since we may assume that $K\delta \leqslant 1/2$). In order to facilitate a union bound argument, we will estimate the size of $\mathcal{S}(s,m,b)$ with small b and m using Proposition 7. To this end, we first argue that each set in $\mathcal{S}(s,m,b)$ can be exactly described by the following:

1. a set $W = \{v_1, \ldots, v_m\}$ of m vertices of G,
2. a partition $s = s_1 + \ldots + s_m$, where $s_i \geqslant 1$ for each i,
3. a partition $b = b_1 + \ldots + b_m$, where $b_i \geqslant 1$ for each i, and
4. a set S_i in $\mathcal{C}(v_i, s_i, b_i)$ for each $i \in [m]$.

To see this, note that we may assume that there is a canonical linear ordering on the vertices of G. The representation of S in $\mathcal{S}(s,m,b)$ as (i)–(iv) is natural. Indeed, given such an S, we find the unique partition $\{S_1, \ldots, S_m\}$ into connected components of $G[S]$ and arbitrarily choose one vertex from each S_i to form W. We order the sets $S_1, \ldots, S_m$ according to the canonical linear ordering on their representatives $v_1, \ldots, v_m$. Finally, we let $s_i = |S_i|$ and $b_i = |\partial_G S_i|$. Observe that this mapping is not only injective, but actually each set S can be represented in $s_1 \cdot \ldots \cdot s_m$ different ways.

It follows from Proposition 7, as well as from the inequality $\binom{x}{y}\binom{w}{z} \leqslant \binom{x+w}{y+z}$, that

$$\begin{aligned}
|\mathcal{S}(s,m,b)| &\leqslant \binom{n}{m} \sum_{(s_i),(b_i)} \prod_{i=1}^{m} \binom{s_i + b_i - 1}{b_i} \leqslant \binom{n}{m} \sum_{(s_i),(b_i)} \binom{\sum_i (s_i + b_i - 1)}{\sum_i b_i} \\
&\leqslant \binom{n}{m}\binom{s-1}{m-1}\binom{b-1}{m-1}\binom{s+b-m}{b} \leqslant \binom{n}{m}\binom{s}{m}\binom{b}{m}\binom{s+b}{b}.
\end{aligned}$$

Consequently, if $m \leqslant b \leqslant \delta s / \log(en/s) \leqslant s$, then it follows from the well-known estimate $\binom{x}{y} \leqslant \left(\frac{ex}{y}\right)^y$ and the fact that the function $y \mapsto (ex/y)^y$ is increasing on the interval $(0, x]$ that

$$\begin{aligned}
|\mathcal{S}(s,m,b)| &\leqslant \left(\frac{en}{m}\right)^m \left(\frac{es}{m}\right)^m \left(\frac{eb}{m}\right)^m \left(\frac{e(s+b)}{b}\right)^b \leqslant \left(\frac{e^4 n s (s+b)}{b^3}\right)^b \\
&\leqslant \left(\frac{2e^4 n (\log(en/s))^3}{\delta^3 s}\right)^{\frac{\delta s}{\log(en/s)}} \leqslant \exp\left(C\delta \log(1/\delta) s\right),
\end{aligned}$$

where C is some absolute constant.

On the other hand, by Chernoff's inequality, for a fixed set S with $|S| = s \leqslant \alpha n$,

$$\begin{aligned}
\Pr(|\partial_R S| < \delta s) &\leqslant \Pr\big(\mathrm{Bin}(s(n-s), \varepsilon/n) < \delta s\big) \\
&\leqslant \Pr\big(\mathrm{Bin}((1-\alpha)sn, \varepsilon/n) < \delta s\big) \leqslant \exp\big(-(1-\alpha)\varepsilon s/8\big).
\end{aligned}$$

provided that $\delta < (1-\alpha)\varepsilon/2$.

Finally, choose positive constants K and δ such that

$$K \geqslant \frac{64}{\varepsilon(1-\alpha)}, \qquad C\delta \log(1/\delta) \leqslant \frac{\varepsilon(1-\alpha)}{16}, \qquad \text{and} \qquad K\delta \leq 1/2.$$

Taking a union bound over all triples b, m, and s satisfying $K \log n \leqslant s \leqslant \alpha n$ and $m \leqslant b \leqslant \delta s / \log(en/s)$, we get that

$$\Pr(\text{property (1) fails}) \leqslant n^3 \exp\left[\left(C\delta \log(1/\delta) - \frac{\varepsilon(1-\alpha)}{8}\right) s\right] = o(1). \qquad \blacktriangleleft$$

4 Diameter

In this section, we prove Theorem 4. Since adding edges to a graph can only decrease its diameter, it suffices to consider the case when G is a tree and $\varepsilon \leqslant 1/3$. Since $e(G) = n - 1$, it follows from Chernoff's inequality (Lemma 9) that a.a.s. G^* has at most $(1+\varepsilon)n$ edges. Hence, it is enough to prove that there is a constant $C = C(\varepsilon)$ such that a.a.s.

$$e\big(B(v, C\log n)\big) > \frac{(1+\varepsilon)n}{2} \qquad \text{for every } v \in V(G), \tag{2}$$

where $B(v,r)$ denotes the G^*-ball of radius r around v. Indeed, (2) implies that for every $u, v \in V(G)$, we have that $B(u, C\log n) \cap B(v, C\log n) \neq \emptyset$, and consequently $\operatorname{diam}(G^*) \leqslant 2C \log n$.

Fix some $v \in V(G)$, let K and δ be as in Theorem 3 with $\alpha = 3/4$, and condition on the event that G^* satisfies the assertion of this theorem. Moreover, condition on the event that R satisfies the assertion of Lemma 8 with $C = 1$. This implies that $e(S)/3 \leqslant |S| \leqslant e(S) + 1$ for every connected set S in G^*. Since G^* is connected, we clearly have that $|B(v,r)| \geqslant r + 1$. Hence, if $r \geqslant K \log n$, we have that

$$e(B(v, r+1)) \geqslant \min\left\{\frac{3n}{4} - 1, \left(1 + \frac{\delta}{3}\right) e\big(B(v,r)\big)\right\}.$$

Letting $C = K + \frac{1}{\log(1+\delta/3)}$, we have that

$$e(B(v, C\log n)) \geqslant \frac{3n}{4} - 1 > \frac{2n}{3} \geqslant \frac{(1+\varepsilon)n}{2},$$

as claimed. $\blacktriangleleft$

5 Mixing Time

In this section, we prove Theorem 5. Let ε be a positive real, let D be a positive integer, and assume that G is a connected D-degenerate graph. Let $G^* = G \cup R$, where $R \sim G(n, \varepsilon/n)$.

Our argument for bounding the mixing time is based on the approach of Fountoulakis and Reed [16, 17]. The main idea there is that one can bound the mixing time of an abstract irreducible, reversible, and aperiodic Markov chain in terms of the *conductances* of *connected* sets of states of various sizes. For simplicity, we only state their results in the setting of the lazy random walk on the graph G^*. For $S \subseteq V$, let $\pi(S)$ equal $\sum_{v \in S} \pi(v)$. It can be verified that $\pi(S) = \frac{2e_{G^*}(S) + |\partial_{G^*} S|}{2e(G^*)}$. We define

$$Q(S) = \sum_{u \in S, v \notin S} \pi(u) \Pr(u \to v) = \frac{|\partial_{G^*} S|}{4e(G^*)}$$

and note that $Q(S) = Q(S^c)$. The *conductance* $\Phi(S)$ of S is

$$\Phi(S) = \frac{Q(S)}{\pi(S)\pi(S^c)} = \frac{|\partial_{G^*} S|}{2 \cdot (2e_{G^*}(S) + |\partial_{G^*} S|) \cdot \pi(S^c)}.$$

Let $\pi_{\min} = \min_{v \in V(G)} \pi(v)$. For $p > \pi_{\min}$, we denote by $\Phi(p)$ the minimum conductance of a connected (in G^*) set S with $p/2 \leqslant \pi(S) \leqslant p$ (if there is no such S, we define $\Phi(p) = 1$). Fountoulakis and Reed [16] proved the following result.

▶ **Theorem 10.** *There exists an absolute constant C such that*

$$T_{\mathrm{mix}}(G^*) \leqslant C \sum_{j=1}^{\lceil \log_2 \pi_{\min}^{-1} \rceil} \Phi^{-2}(2^{-j}).$$

In the remainder of the proof, we will estimate the sum in Theorem 10. We claim that it is enough to prove the following.

▶ **Lemma 11.** *There exist positive constants δ^* and K^* such that a.a.s. for every connected (in G^*) set S with $\frac{K^* \log n}{n} \leqslant \pi(S) \leqslant 1/2$,*

$$\Phi(S) \geqslant \delta^*.$$

Indeed, suppose that the assertion of Lemma 11 holds for some δ^* and K^*. Let J be the set of indices j satisfying $2^{-j} \leqslant \frac{2K^* \log n}{n}$ and note the $|J^c| < \log_2 n$, as $2^{-j} > \frac{2K^* \log n}{n}$ implies that $j < \log_2 n$. Since G^* is connected, we have that for every set S,

$$\Phi(S) \geqslant \frac{|\partial_{G^*} S|}{4e(G^*) \cdot \pi(S)} \geqslant \frac{1}{4e(G^*) \cdot \pi(S)}.$$

Condition on the event that R satisfies the assertion of Lemma 8 with $C = \max\{\varepsilon, 1\}$. Let $D^* = D + 2C$ and observe that the degeneracy assumption implies that

$$e_{G^*}(S) \leqslant D^*|S| \quad \text{for every } S \subseteq V(G). \tag{3}$$

In particular, $e(G^*) \leqslant D^* n$ and hence, letting $M = 129(K^*)^2(D^*)^2$,

$$\begin{aligned}\sum_{j=1}^{\lceil \log_2 \pi_{\min}^{-1} \rceil} \Phi^{-2}(2^{-j}) &\leqslant |J^c| \cdot (\delta^*)^{-2} + \sum_{j \in J} 2^{-2j}(4e(G^*))^2 \\ &\leqslant O(\log n) + 2 \cdot \max_{j \in J}\{2^{-2j}\} \cdot 16(D^*)^2 n^2 \leqslant M \log^2 n,\end{aligned}$$

provided that n is sufficiently large, where we used the definition of J and the inequality $\sum_{j \geqslant i} 2^{-2j} \leqslant 2^{-2i+1}$.

Therefore, it suffices to prove Lemma 11. We first show that any connected set S with $\pi(S) \leqslant 1/2$ has at most $n - \Omega(n)$ elements.

▶ **Claim 12.** *Every connected (in G^*) set $S \subseteq V(G)$ with $\pi(S) \leqslant 1/2$ satisfies*

$$|S| \leqslant \frac{D^* n + 1}{D^* + 1}.$$

Proof. Since $\pi(S) \leqslant 1/2$ implies that $\pi(S) \leqslant \pi(S^c)$, we have

$$\begin{aligned}2e_{G^*}(S) &= 2e(G^*)\pi(S) - |\partial_{G^*} S| \leqslant 2e(G^*)\pi(S^c) - |\partial_{G^*} S| \\ &= 2e_{G^*}(S^c) \leqslant 2D^*|S^c| = 2D^*(n - |S|).\end{aligned}$$

Since S is connected in G^*, we obtain $e_{G^*}(S) \geqslant |S| - 1$ and the claim follows. ◀

Let δ and K be as in Theorem 3 with $\alpha = \frac{D^*}{D^*+1} + 1$ and condition on the event that G^* satisfies the assertion of this theorem. Let

$$K^* = (2D^* + 1)K \qquad \text{and let} \qquad \delta^* = \min\left\{\frac{1}{2D^*+2}, \frac{\delta}{2D^*+2\delta}\right\}.$$

It follows from (3) that for every connected set S with $\pi(S) \leqslant 1/2$, we have

$$\Phi(S) \geqslant \frac{|\partial_{G^*} S|}{2 \cdot (e_{G^*}(S) + |\partial_{G^*} S|)} \geqslant \frac{|\partial_{G^*} S|}{2D^*|S| + 2|\partial_{G^*} S|}.$$

Note that if $|\partial_{G^*} S| \geqslant |S|$, then $\Phi(S) \geqslant \delta^*$, so we may assume otherwise. In particular, if $\pi(S) \geqslant K^* \log n/n$, then as $e(G^*) \geq n$ we have

$$K^* \log n \leqslant 2\pi(S)e(G^*) = 2e_{G^*}(S) + |\partial_{G^*} S| \leqslant (2D^* + 1)|S|$$

and hence $|\partial_{G^*} S| \geqslant \delta|S|$. It follows that $\Phi(S) \geqslant \delta^*$. This concludes the proof of Lemma 11 and therefore the proof of Theorem 5. ◀

6 Concluding Remarks

In this paper, we studied the model of randomly perturbed connected graphs. Using a new general upper bound on the number of connected subsets with small vertex boundary, we proved lower bounds on edge expansion under mild assumptions on the base graph. We established several other interesting properties of randomly perturbed connected graphs: bounds on the diameter and the mixing time of the lazy random walk. It would be interesting to study other parameters of this model.

It seems that randomly perturbed connected n-vertex graph with bounded degeneracy shares some similarities with the giant component in the supercritical Erdős-Rényi random graph $G(n, \frac{1+\varepsilon}{n})$. In particular, a.a.s. they both have diameter $O(\log n)$, mixing time $O(\log^2 n)$, and contain paths of length $\Omega(n)$. It could be interesting to explore this analogy further and to check whether the methods used in this work to study the model of randomly perturbed graphs can be applied to the other model.

Acknowledgments. We would like to thank Uri Feige, Jon Kleinberg, Gady Kozma, and Ofer Zeitouni for motivating discussions. We thank Yuval Peres for pointing out an inaccuracy in a previous version of this work.

References

1 L. Addario-Berry and T. Lei. The mixing time of the Newman–Watts small world. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'12, pages 1661–1668, 2012.

2 M. Ajtai, J. Komlós, and E. Szemerédi. The longest path in a random graph. *Combinatorica*, 1:1–12, 1981.

3 I. Benjamini, G. Kozma, and N. Wormald. The mixing time of the giant component of a random graph. arXiv:math/0610459 [math.PR].

4 I. Benjamini and E. Mossel. On the mixing time of a simple random walk on the super critical percolation cluster. *Probab. Theory Related Fields*, 125:408–420, 2003.

5 A. Blum and J. Dunagan. Smoothed analysis of the perceptron algorithm for linear programming. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'02, pages 905–914, 2002.

6 A. Blum and J. Spencer. Coloring random and semi-random k-colorable graphs. *J. Algorithms*, 19:204–234, 1995.

7 F. Bohman, A. Frieze, and R. Martin. How many random edges make a dense graph hamiltonian? *Random Structures Algorithms*, 22:33–42, 2003.

8 B. Bollobás and F. R. Chung. The diameter of a cycle plus a random matching. *SIAM J. Discrete Math.*, 1:328–333, 1988.

9 A. Coja-Oghlan, U. Feige, A. Frieze, M. Krivelevich, and D. Vilenchik. On Smoothed k-CNF Formulas and the Walksat Algorithm. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'09, pages 451–460, 2009.

10 J. Ding and Y. Peres. Sensitivity of mixing times. *Electron. Commun. Probab.*, 18:1–6, 2013.

11 R. Durrett. *Random graph dynamics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge, 2010.

12 U. Feige. Refuting Smoothed 3CNF Formulas. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS' 07, pages 407–417, 2007.

13 U. Feige and J. Kilian. Heuristics for semirandom graph problems. *J. Comput. System Sci.*, 63:639–671, 2001.

14 A. Flaxman and A. Frieze. The diameter of randomly perturbed digraphs and some applications. *Random Structures Algorithms*, 30:484–504, 2007.

15 A. D. Flaxman. Expansion and lack thereof in randomly perturbed graphs. *Internet Math.*, 4:131–147, 2007.

16 N. Fountoulakis and B. A. Reed. Faster mixing and small bottlenecks. *Probab. Theory Related Fields*, 137:475–486, 2007.

17 N. Fountoulakis and B. A. Reed. The evolution of the mixing rate of a simple random walk on the giant component of a random graph. *Random Structures Algorithms*, 33:68–86, 2008.

18 M. Jerrum and A. Sinclair. Conductance and the rapid mixing property for markov chains: the approximation of the permanent resolved. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, STOC'88, pages 235–244, 1988.

19 M. Krivelevich and A. Nachmias. Coloring complete bipartite graphs from random lists. *Random Structures Algorithms*, 29:436–449, 2006.

20 M. Krivelevich and B. Sudakov. The phase transition in random graphs – a simple proof. *Random Structures Algorithms*, 43:1–15, 2013.

21 M. Krivelevich, B. Sudakov, and P. Tetali. On smoothed analysis in dense graphs and formulas. *Random Structures Algorithms*, 29:180–193, 2006.

22 D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov chains and mixing times*. American Mathematical Society, Providence, RI, 2009.

23 M. Mitzenmacher and E. Upfal. *Probability and Computing, Randomized algorithms and probabilistic analysis*. Cambridge University Press, Cambridge, 2005.

24 M. E. J. Newman and D. J. Watts. Renormalization group analysis of the small-world network model. *Phys. Lett. A*, 263:341–346, 1999.

25 M. E. J. Newman and D. J. Watts. Scaling and percolation in the small-world network model. *Phys. Rev. E (3)*, 60:7332–7342, 1999.

26 A. Sankar, D. Spielman, and S. H. Teng. Smoothed analysis of the condition numbers and growth factors of matrices. *SIAM J. Matrix Anal. Appl.*, 28:446–476, 2006.

27 J. Spencer and G. Tóth. Crossing numbers of random graphs. *Random Structures Algorithms*, 21:347–358, 2002.

28 D. Spielman and S. H. Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM*, 51:385–463, 2004.

29 B. Sudakov and J. Vondrák. How many random edges make a dense hypergraph non-2-colorable? *Random Structures Algorithms*, 32:290–306, 2008.

30 T. Tao and V. Vu. The condition number of a randomly perturbed matrix. In *STOC'07—Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 248–255. ACM, New York, 2007.

A Vertex Expansion

Here we prove Theorem 1.

Proof of Theorem 1. Let

$$k = C\Delta \log n,$$

where $C = C(\varepsilon)$ is a large enough constant, which we will define later. Partition the vertex set of G into *disjoint* pieces $V_1, \ldots, V_t$, such that for each $1 \leqslant i \leqslant t$, we have $k \leqslant |V_i| \leqslant \Delta k$ and $G[V_i]$ is *connected*. This is fairly straightforward, see, e. g., [19, Proposition 4.5]. Observe that

$$\frac{n}{\Delta k} \leqslant t \leqslant \frac{n}{k}.$$

Call each V_i a *blob*.

The probabilistic statement about the random graph R we need is the following one: a.a.s. for every non-empty $I \subseteq [t]$ with $|I| \leqslant t/2$, there are at least $|I|/2$ blobs with indices outside of I that are connected by an edge to the set $\bigcup_{i\in I} V_i$. Let

$$\rho := \frac{\varepsilon k^2}{2n}.$$

Clearly, the probability that two blobs are connected in $G \cup R$ is at least $1-(1-p)^{|V_i|\cdot|V_j|} \geqslant \rho$ (if the two blobs are connected in G, then this probability is 1). The probability P that there exists a non-empty set $I \subseteq [t]$ with $|I| \leqslant t/2$ such that the set $\bigcup_{i\in I} V_i$ has an edge (in R) to fewer than $|I|/2$ blobs outside of I satisfies

$$P \leqslant \sum_{1\leqslant j\leqslant t/2} \binom{t}{j}\binom{t-j}{j/2}(1-\rho)^{(t-\frac{3j}{2})j} \leqslant \sum_{1\leqslant j\leqslant t/2} t^{\frac{3j}{2}+1} \cdot \exp\left(-\frac{\rho j t}{4}\right).$$

It is easy to verify that $P = o(1)$ if we take $C(\epsilon) \geqslant C'/\varepsilon$ for a sufficiently large absolute constant C'.

Suppose that R has the above property. We claim that $G \cup R$ has vertex expansion at least $\frac{\delta}{\Delta^3 \log n}$ for some positive constant $\delta = \delta(\varepsilon)$. Fix a set $A \subseteq [n]$ with $|A| \leqslant n/2$ and denote

$$I_0 = I_0(A) = \{1 \leqslant i \leqslant t \colon V_i \subseteq A\},$$
$$I_1 = I_1(A) = \{1 \leqslant i \leqslant t \colon \emptyset \neq V_i \cap A \neq V_i\},$$
$$I_2 = I_2(A) = \{i \notin I_0 \colon V_i \text{ has a neighbor in } A\}.$$

In other words, I_0 is the set of (indices of) blobs fully contained in A, I_1 is the set of blobs having at least one vertex in A but not falling completely inside A, and finally I_2 is the set of blobs outside I_0 having a neighbor in A.

It follows from the above stated property of R that a.a.s.

$$|I_2| \geqslant \min\left\{\frac{|I_0|}{2}, \frac{t-|I_0|}{3}\right\}.$$

This is clear when $|I_0| \leqslant t/2$; we simply take $I = I_0$. Else, we let $I = [t] \setminus (I_0 \cup I_2)$, note that $|I| \leqslant t - |I_0| \leqslant t/2$, and observe that no blob in I is connected to a blob in I_0 (and hence the neighborhood of I must be completely contained in I_2). Observe crucially that $|N(A)| \geqslant |I_1|$ since each set $V_i \cap A$ with $i \in I_1$ has at least one neighbor in $V_i \setminus A$ due to the connectivity of the blob $G[V_i]$. Also, $|N(A)| \geqslant |I_2 \setminus I_1|$ as for every blob V_i with $i \in I_2 \setminus I_1$, A has a neighbor in $V_i \setminus A$. Finally, note that $A \subseteq \bigcup_{i \in I_0 \cup I_1} V_i$, implying that

$$|I_0| + |I_1| \geqslant \frac{|A|}{\Delta k}. \tag{4}$$

If $|I_1| \geqslant \frac{|I_0|}{6\Delta}$, then it follows from (4) that $|I_1| \geqslant \frac{|A|}{\Delta(1+6\Delta)k}$ and thus $|N(A)| \geqslant |I_1| \geqslant \frac{|A|}{\Delta(1+6\Delta)k}$. If $|I_1| \leqslant \frac{|I_0|}{6\Delta}$, then we distinguish between two cases, depending on the value of $\min\{\frac{|I_0|}{2}, \frac{t-|I_0|}{3}\}$. If $\frac{|I_0|}{2} \leqslant \frac{t-|I_0|}{3}$, then $|I_2 \setminus I_1| \geqslant \frac{|I_0|}{2} - \frac{|I_0|}{6\Delta} \geq \frac{|I_0|}{3}$. As $|N(A)| \geqslant |I_2 \setminus I_1|$, we get by (4) and our assumption on $|I_1|$ that $|N(A)| \geqslant \frac{|I_0|}{3} \geqslant \frac{(1+1/(6\Delta))|A|}{3\Delta k}$. Else, if $\frac{t-|I_0|}{3} < \frac{|I_0|}{2}$, we observe that, as $|A| \leqslant \frac{n}{2}$ and $A^c \subseteq \bigcup_{i \notin I_0} V_i$, we have $|I_0| \leqslant (1 - \frac{1}{\Delta})t$. In this case, $|I_2 \setminus I_1| \geqslant \frac{t-|I_0|}{3} - \frac{|I_0|}{6\Delta}$ and hence $|N(A)| \geqslant \frac{t}{6\Delta} \geqslant \frac{|A|}{6\Delta^2 k}$. Hence, G^* has vertex expansion at least $\frac{\delta}{\Delta^3 \log n}$, where $\delta = \delta'\varepsilon$ and δ' is an absolute positive constant. ◀

Remark. Observe that the exact same proof as above works if instead of assuming that the graph G is connected and has maximum degree bounded by Δ, we assume that $\Delta(G) \leqslant \Delta$ and all connected components of G are at least as large as $C\Delta \log n$, where $C = C(\epsilon)$ is a large enough constant.

It is natural to ask what happens when $\epsilon(n)$ tends to zero with n. Similar ideas to those used in our proof of Theorem 1 apply in this case as well. We illustrate this in the case when $\varepsilon = n^{-a}$ for some $a \in (0,1)$. Observe that if $p = O(\frac{1}{n^2})$, then a.a.s. the number of random edges that are added is constant, hence if G is a tree, then a.a.s. G^* has expansion $O(\frac{1}{n})$. As the proof closely follows the lines of the above proof of Theorem 1, we only give an outline of the arguments, omitting some of the details.

▶ **Proposition 13.** *Let G be an n-vertex connected graph with maximum degree Δ and set $\varepsilon = n^{-a}$ for some $a \in (0,1)$. If $R \sim G(n, \frac{\varepsilon}{n})$, then a.a.s. the graph $G^* = G \cup R$ has vertex expansion at least $\Omega(\frac{1}{\log n \cdot n^a \cdot \Delta^3})$.*

Proof. Let $k = C\Delta \log n \cdot n^a$ where C is a sufficiently large constant. Partition the vertex set of G into disjoint connected blobs with each blob of size between k and Δk. The number of blobs t again satisfies $\frac{n}{\Delta k} \leqslant t \leqslant \frac{n}{k}$. Similar arguments to those in the proof of Theorem 1 show that a.a.s. for every set I of size at most $t/2$, there are at least $|I|/2$ blobs outside of I that are connected by an edge to the set $\bigcup_{i \in I} V_i$. This implies, as before, that a.a.s. the expansion of G^* is $\Omega(\frac{1}{\log n \cdot n^a \cdot \Delta^3})$. ◀

B Proof of Theorem 3

Here we prove Theorem 3.

Proof. Due to the obvious monotonicity we can assume that $\epsilon < 1$. Recall the definition of $\mathcal{S}(s,m,b)$ from the proof of Theorem 2. It clearly suffices to show that a.a.s. for every s with $K \log n \leqslant s \leqslant \alpha n$,

$$|\partial_R S| \geqslant \delta s \quad \text{for all } \textit{connected } S \in \mathcal{S}(s,m,b) \text{ with } m \leqslant b < \delta s, \tag{5}$$

where connected means connected in the graph G^*.

Let us denote by $\mathcal{S}'(s,m,b)$ the collection of all ordered pairs

- $S = S_1 \cup \ldots \cup S_m \in \mathcal{S}(s,m,b)$, where $S_1, \ldots, S_m$ are connected components of $G[S]$,
- $m-1$ pairs of vertices of S, $s_1, t_1, \ldots, (s_{m-1}, t_{m-1})$ such that adding the edges $(s_1,t_1), \ldots, (s_{m-1}, t_{m-1})$ to the edges G makes $G[S]$ connected.

A moment of thought reveals that for fixed s, the probability that (5) does not hold is bounded by

$$\sum_{m=1}^{\delta s} \sum_{b=m}^{\delta s} |\mathcal{S}'(s,m,b)| \cdot (\varepsilon/n)^{m-1} \cdot \Pr\big(\mathrm{Bin}(s(n-s), \varepsilon/n) \leqslant \delta s\big). \tag{6}$$

Therefore, it suffices to prove the following.

▶ **Claim 14.** *There exists an absolute constant C such that for all s, m, and b with $m \leqslant b \leqslant \delta s$,*

$$|\mathcal{S}'(s,m,b)| \leqslant n^m \exp(C\delta \log(1/\delta) s).$$

Indeed, if $K \log n \leqslant s \leqslant \alpha n$, then by Chernoff's inequality,

$$\begin{aligned}\Pr\big(\mathrm{Bin}(s(n-s), \varepsilon/n) < \delta s\big) &\leqslant \Pr\big(\mathrm{Bin}((1-\alpha)sn, \varepsilon/n) < \delta s\big) \\ &\leqslant \exp\big(-(1-\alpha)\varepsilon s/8\big),\end{aligned}$$

provided that $\delta < (1-\alpha)\varepsilon/2$. Hence, (6) is bounded from above by

$$s^2 n \exp\left[\left(C\delta \log(1/\delta) - \frac{(1-\alpha)\varepsilon}{8}\right) s\right].$$

If we choose K and δ as in the proof of Theorem 2, a union bound over $K \log n < s \leqslant \alpha n$ yields that (6) is indeed $o(1)$.

Hence, it suffices to prove the claim. To this end, we will argue that each element of $\mathcal{S}'(s,m,b)$ can be uniquely described by the following:

1. a set $W = \{v_1, \ldots, v_m\}$ of vertices of G,
2. a partition $s = s_1 + \ldots + s_m$, where $s_i \geqslant 1$ for each i,
3. a partition $b = b_1 + \ldots + b_m$, where $b_i \geqslant 1$ for each i,
4. a set S_i in $\mathcal{C}(v_i, s_i, b_i)$ for each $i \in [m]$,
5. a partition $m - 1 = d_1 + \ldots + d_m$, where $d_i \geqslant 0$ for each i,
6. a multiset D_i of d_i elements from S_i for each $i \in [m]$,
7. a permutation $f\colon [m-1] \to [m-1]$.

Assuming that this is indeed the case, by Proposition 7 we have

$$\begin{aligned}|\mathcal{S}'(s,m,b)| &\leqslant \binom{n}{m}(m-1)! \sum_{(s_i),(b_i),(d_i)} \prod_{i=1}^{m} \left[\binom{s_i + b_i - 1}{b_i}\binom{s_i + d_i - 1}{d_i}\right] \\ &\leqslant \frac{n^m}{m}\binom{s-1}{m-1}^2 \binom{b-1}{m-1}\binom{2m-2}{m-1}\binom{s+b-m}{b} \leqslant (2n)^m \binom{s}{m}^2 \binom{b}{m}\binom{s+b}{b}.\end{aligned}$$

Consequently, if $m \leqslant b \leqslant \delta s$, then

$$|\mathcal{S}'(s,m,b)| \leqslant n^m \left(\frac{2e^4 s^2(s+b)}{b^3}\right)^b \leqslant n^m \left(\frac{3e^4}{\delta^3}\right)^{\delta s} \leqslant n^m \exp\left(C\delta \log(1/\delta) s\right),$$

where C is some absolute constant.

Finally, we show that each $S \in \mathcal{S}'(s,m,b)$ may be uniquely described by (i)–(vii). First, observe that (i)–(iv) uniquely describe the set $S = S_1 \cup \ldots \cup S_m$, together with a root vertex v_i in each connected component S_i, whose use will be explained later. As in the proof of Theorem 2, one may assume some canonical linear ordering $\preceq$ on the set of vertices of G. Given this ordering, one may canonically order the sets $S_1, \ldots, S_m$ according to the canonical ordering $\preceq$ on the set $\{\min_{\preceq} S_1, \ldots, \min_{\preceq} S_m\}$ of representatives of each S_i. Now, note that the $m-1$ pairs of vertices of S whose addition to G makes $G[S]$ connected naturally define a tree T on the vertex set $\{S_1, \ldots, S_m\}$. Root this tree at S_m and orient all of its edges away from the root. Now, start with $v_m \in S_m$ and for each $i \in [m-1]$ let $v_i \in S_i$ be the unique vertex of S_i that lies in the pair of vertices of S that corresponds to the unique edge of T going into S_i. Next, for each $i \in [m]$, let d_i be the outdegree of S_i in T and let D_i be the multiset of d_i vertices of S_i that lie in the pairs of vertices of S that correspond to the d_i edges of T going out of S_i. Finally, let $D = D_1 \cup \ldots \cup D_m$ and observe that the $m-1$ pairs of vertices of S that correspond to the edges of T define a bijection between D and $\{v_1, \ldots, v_{m-1}\}$. Namely, if $D = \{w_1, \ldots, w_{m-1}\}$, where $w_1 \preceq \ldots \preceq w_{m-1}$, then this bijection can be described by a permutation $f\colon [m-1] \to [m-1]$ defined by letting $f(i)$ be the unique j such that $\{v_i, w_j\}$ is one of the $m-1$ pairs of vertices whose addition to G makes $G[S]$ connected. This concludes the proof of the theorem. ◀

Here we show that after adding random edges, each with probability $\frac{\epsilon}{n}$, to a connected n-vertex graph with bounded maximum degree, we a.a.s. get a path whose length is linear in n.

Proof of Theorem 6. Let k be a sufficiently large constant. Similarly as in the proof of Theorem 1, let us chop the vertex set of the graph G into connected pieces $V_1, \ldots, V_t$ such that for each $1 \leqslant i \leqslant t$, we have $k \leqslant |V_i| \leqslant \Delta k$. As in Theorem 1, the probability that two blobs are connected (in G^*) is at least $\frac{\varepsilon k^2}{2n}$. Hence, if k is sufficiently large, then the auxiliary graph naturally induced by the blobs (obtained by treating each blob as a super-vertex and connecting two super-vertices if there is an edge of G^* connecting the two blobs) contains the random graph $G(t, C/t)$, where $C \to \infty$ as $k \to \infty$. It is well known ([2], see also [20]) that if $C > 1$, then $G(t, C/t)$ a.a.s. contains a path P_0 of length $\Omega(t)$. Since the blobs are connected and $t \geqslant n/(\Delta k)$, one can turn P_0 into a path P in G^*, whose length is at least as large as the length of P_0. Indeed, we may use the edges of P_0 to move between the blobs and the edges of G to connect the entry and the exit points of P_0 within each blob traversed by P_0. ◀

Local Algorithms for Sparse Spanning Graphs*

Reut Levi[1], Dana Ron[2], and Ronitt Rubinfeld[3]

1 **School of Computer Science, Tel Aviv University**
Tel Aviv 69978, Israel
reuti.levi@gmail.com
2 **School of Electrical Engineering, Tel Aviv University**
Tel Aviv 69978, Israel
danar@eng.tau.ac.il
3 **CSAIL, MIT**
Cambridge MA 02139, USA; and
School of Electrical Engineering, Tel Aviv University
Tel Aviv 69978, Israel
ronitt@csail.mit.edu

Abstract

We initiate the study of the problem of designing sublinear-time (*local*) algorithms that, given an edge (u, v) in a connected graph $G = (V, E)$, decide whether (u, v) belongs to a sparse spanning graph $G' = (V, E')$ of G. Namely, G' should be connected and $|E'|$ should be upper bounded by $(1 + \epsilon)|V|$ for a given parameter $\epsilon > 0$. To this end the algorithms may query the incidence relation of the graph G, and we seek algorithms whose query complexity and running time (per given edge (u, v)) is as small as possible. Such an algorithm may be randomized but (for a fixed choice of its random coins) its decision on different edges in the graph should be consistent with the same spanning graph G' and independent of the order of queries.

We first show that for general (bounded-degree) graphs, the query complexity of any such algorithm must be $\Omega(\sqrt{|V|})$. This lower bound holds for graphs that have high expansion. We then turn to design and analyze algorithms both for graphs with high expansion (obtaining a result that roughly matches the lower bound) and for graphs that are (strongly) non-expanding (obtaining results in which the complexity does not depend on $|V|$). The complexity of the problem for graphs that do not fall into these two categories is left as an open question.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases local, spanning graph, sparse

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.826

1 Introduction

When dealing with large graphs, it is often important to work with a sparse subgraph that maintains essential properties, such as connectivity, bounded diameter and other distance metric properties, of the original input graph. Can one provide fast random access to such a sparsified approximation of the original input graph? In this work, we consider the property of connectivity: Given a connected graph $G = (V, E)$, find a sparse subgraph of G' that spans G. This task can be accomplished by constructing a spanning tree in linear time. However, it can be crucial to *quickly* determine whether a particular edge e belongs to such

* This work was partially supported by the Israel Science Foundation grant number nos. 1147/09, 246/08, and 671/13 and by NSF grants CCF-1217423 and CCF-1065125.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 826–842

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

a subgraph G', where by "quickly" we mean in time much faster than constructing all of G'. The hope is that by inspecting only some small local neighborhood of e, one can answer in such a way that maintains consistency with the same G' on queries to all edges. We focus on such algorithms, which are of use when we do not need to know the answer for every edge at any single point in time, or if there are several independent processes that want to determine the answer for edges of their choice, possibly in parallel.

If we insist that G' would have the *minimum* number of edges sufficient for spanning G, namely, that G' be a spanning tree, then it is easy to see that the task cannot be performed in general without inspecting almost all of G. Interestingly, this is in contrast to the seemingly related problem of estimating the weight of a minimum spanning tree in sublinear-time, which can be performed with complexity that does not depend on $n \stackrel{\text{def}}{=} |V|$ [11] (see further discussion in Subsection 1.3.5). To verify this observe that if G consists of a single path, then the algorithm must answer positively on all edges, while if G consists of a cycle, then the algorithm must answer negatively on one edge. However, the two cases cannot be distinguished without inspecting a linear number of edges. If on the other hand we allow the algorithm some more slackness, and rather than requiring that G' be a tree, require that it be relatively sparse, i.e., contains at most $(1+\epsilon)n$ edges, then the algorithm may answer positively on all cycle edges, so distinguishing between these two cases is no longer necessary.

We thus consider the above relaxed version of the problem and also allow the algorithm a small failure probability (for a precise formal definition, see Section 2). Our first finding (Theorem 2) is that even when allowing this relaxation, for general (bounded-degree) graphs, the algorithm must inspect $\Omega(\sqrt{n})$ edges in G in order to decide for a given e whether it belongs to the sparse spanning graph G' defined by the algorithm. We then turn to design several algorithms and analyze their performance for various families of graphs. The formal statements of our results can be found in Theorems 3, 4, 5, 6, and 7 as well as Corollaries 12 and 13. Here we provide a high-level description of our algorithms and the types of graphs they give meaningful results for.

1.1 Our Results

1.1.1 Expanders

The first algorithm we provide, the Centers' Algorithm (which is discussed further in Subsection 1.2), gives meaningful results for graphs in which the neighborhoods of almost all the vertices in the graph expands in a similar rate. In particular, for graphs with high expansion we get query and running time complexity nearly $O(n^{1/2})$. Since our lower bound applies for graphs with high expansion we obtain that for these graphs, our algorithm is nearly optimal in terms of the complexity in n. More specifically, if the expansion of small sets (of size roughly $O(n^{1/2})$) is $\Omega(d)$, where d is the maximum degree in the graph, then the complexity of the algorithm is $n^{1/2+O(1/\log d)}$. In general, we obtain a sublinear complexity for graphs with expansion (of small sets) that is at least $d^{1/2+1/\log n}$.

1.1.2 Anti-expanders (Hyperfinite Graphs) and Slowly Expanding Graphs

A graph is *ρ-hyperfinite* for a function $\rho : \mathbb{R}_+ \to \mathbb{R}_+$, if its vertices can be partitioned into subsets of size at most $\rho(\epsilon)$ that are connected and such that the number of edges between the subsets is at most ϵn. For the family of hyperfinite graphs we provide an algorithm, the Kruskal-based algorithm, which has success probability 1 and time and query complexity

$O(d^{\rho(\epsilon)})$. In particular, the complexity of the algorithm does not depend on n. (where we assume that $\rho(\epsilon)$ is known).

1.1.2.1 Subfamilies of Hyperfinite Graphs

For the subfamily of hyperfinite graphs known as graphs with subexponential-growth, we can estimate the diameter of the sets in the partition and hence replace $\rho(\epsilon)$ with such an estimate. This reduces the complexity of the algorithm when the diameter is significantly smaller than $\rho(\epsilon)$, and removes the assumption that $\rho(\epsilon)$ is known. For the subfamily of graphs with an excluded minor (e.g., planar graphs) we can obtain a quasi-polynomial dependence on d and $1/\epsilon$ by using a *partition oracle* for such graphs [21], and the same technique gives polynomial dependence on these parameter for bounded-treewidth graphs (applying [14]).

1.1.2.2 Graphs with Slow Growth Rate

If we do not require that the algorithm work for every ϵ but rather for some fixed constant ϵ, then the Kruskal-based algorithm gives sublinear complexity under a weaker condition than that defining (ρ-)hyperfinite graphs (in which the desired partition should exist for every ϵ). Roughly speaking, the sizes of the neighborhoods of vertices should have bounded growth rate, where the rate may be exponential but the base of the exponent should be bounded (for details, see Theorem 7).

1.1.2.3 Graphs with an Excluded Minor – the Weighted Case

In the full version of this paper [23] we provide a local minimum weight spanning graph algorithm, the Borůvka based algorithm, for *weighted* graphs with an excluded fixed minor. The minimum weight spanning graph problem is a generalization of the sparse spanning graph problem for the weighted case. The requirement is that the weight of the graph G' is upper bounded by $(1+\epsilon)$ times the minimum weight of a spanning tree. The time and query complexity of the algorithm are quasi-polynomial in $1/\epsilon$, d and W, where W is the maximum weight of an edge. We use ideas from [21], but the algorithm differs from the abovementioned partition-oracle based algorithm for the unweighted case.

1.2 Our Algorithms

On a high-level, underlying each of our algorithms is the existence of a (global) partition of the graph vertices where edges within parts are dealt with differently than edges between parts, either explicitly by the algorithm, or in the analysis. The algorithms differ in the way the partitions are defined, where in particular, the number of parts in the partition may be relatively small or relatively large, and the subgraphs they induce may be connected or unconnected. The algorithms also differ in the way the spanning graph edges are chosen, and in particular whether only some of the edges between parts are selected or possibly all. While one of our algorithms works in a manner that is oblivious of the partition (and the partition is used only in the analysis), the other algorithms need to determine in a local manner whether the end points of the given edge belong to the same part or not, as a first step in deciding whether the edge belongs to the sparse spanning graph.

Centers' Algorithm. This algorithm is based on the following idea. Suppose we can partition the graph vertices into $\sqrt{\epsilon n}$ disjoint parts where each part is connected. If we now take a spanning tree over each part and select one edge between every two parts that have an

edge between them, then we obtain a connected subgraph with at most $(1+\epsilon)n$ edges. The partition is defined based on $\sqrt{\epsilon n}$ special *center* vertices, which are selected uniformly at random. Each vertex is assigned to the closest center (breaking ties according to a fixed order over the centers), unless it is further than some threshold k from all centers, in which case it is a singleton in the partition. This definition ensures the connectivity of each part.

Given an edge (x, y), the algorithm finds the centers to which x and y are assigned (or determines that they are singletons). If x and y are assigned to the same center, then the algorithm determines whether the edge between them belongs to the BFS-tree rooted at the center. If they belong to different centers, then the algorithm determines whether (x, y) is the single edge allowed to connect the parts corresponding to the two centers (according to a prespecified rule).[1] If one of them is a singleton, then (x, y) is taken to the spanning graph.

From the above description one can see that there is a certain tension between the complexity of the algorithm and the number of edges in the spanning graph G'. On one hand, the threshold k should be such that with high probability (over the choice of the centers) almost all vertices have a center at distance at most k from them. Thus we need a lower bound (of roughly $\sqrt{n/\epsilon}$) on the size of the distance-k neighborhood of (most) vertices. On the other hand, we also need an upper bound on the size of such neighborhoods so that we can efficiently determine which edges are selected between parts. Hence, this algorithm works for graphs in which the sizes of the aforementioned local neighborhoods do not vary by too much and its complexity (in terms of the dependence on n) is $\tilde{O}(\sqrt{n})$. In particular this property holds for good expander graphs. We note that the graphs used in our lower bound construction have this property, so for such graphs we get a roughly tight result.

Kruskal-based Algorithm. This algorithm is based on the well known algorithm of Kruskal [19] for finding a minimum weight spanning tree in a weighted graph. We use the order over the edges that is defined by the ids of their endpoints as (distinct) "weights". This ensures that there is a unique "minimum weight" spanning tree. Here the algorithm simply decides whether to include an edge in the spanning graph G' if it does not find evidence in the distance-k neighborhood of the edge that it is the highest ranking (maximum weight) edge on some cycle.

Borůvka-based Algorithm. This algorithm is based on the "Binary Borůvka" algorithm [36] for finding a minimum-weight spanning tree. Recall that Borůvka's algorithm begins by first going over each vertex in the graph and adding the lightest edge adjacent to that vertex. Then the algorithm continues joining the formed clusters in a similar manner until a tree spanning all vertices is completed. We aim to locally simulate the execution of Borůvka's algorithm to a point that on one hand all the clusters are relatively small and on the other hand the number of edges outside the clusters is small. The size of the clusters directly affect the complexity of the algorithm and thus our main challenge is in maintaining these clusters small. To this end we use two different techniques. The first technique is to control the growth of the clusters at each iteration by using a certain random orientation on the edges of the graph. This controls the size of the clusters to some extent. In order to deal with clusters that exceeded the required bound (since the local simulation is recursive even small deviations can have large impact on the complexity), after each iteration we separate

[1] In fact, it may be the case that for two parts that have edges between them, none of the edges are taken, thus making the argument that the subgraph G' is connected more subtle.

large clusters into smaller ones (here we use the fact that the graph excludes a fixed minor in order to obtain a small separator to each cluster).

1.3 Related Work

1.3.1 Local Algorithms for Other Graph Problems

The model of *local computation algorithms* as used in this work, was defined by Rubinfeld et al. [37] (see also [2]). Such algorithms for maximal independent set, hypergraph coloring, k-CNF and maximum matching are given in [37, 2, 26, 27]. This model generalizes other models that have been studied in various contexts, including locally decodable codes (e.g., [25]), local decompression [13], and local filters/reconstructors [1, 38, 9, 18, 17, 12]. Local computation algorithms that give approximate solutions for various optimization problems on graphs, including vertex cover, maximal matching, and other packing and covering problems, can also be derived from sublinear time algorithms for parameter estimation [33, 28, 31, 15, 41].

Campagna et al. [10] provide a local reconstructor for connectivity. Namely, given a graph which is almost connected, their reconstructor provides oracle access to the adjacency matrix of a connected graph which is close to the original graph. We emphasize that our model is different from theirs, in that they allow the addition of new edges to the graph, whereas our algorithms must provide spanning graphs whose edges are present in the original input graph.

1.3.2 Distributed Algorithms

The name *local algorithms* is also used in the distributed context [29, 30, 24]. As observed by Parnas and Ron [33], local distributed algorithms can be used to obtain local computation algorithms as defined in this work, by simply emulating the distributed algorithm on a sufficiently large subgraph of the graph G. However, while the main complexity measure in the distributed setting is the number of rounds (where it is usually assumed that each message is of length $O(\log n)$), our main complexity measure is the number of queries performed on the graph G. By this standard reduction, the bound on the number of queries (and hence running time) depends on the size of the queried subgraph and may grow exponentially with the number of rounds. Therefore, this reduction gives meaningful results only when the number of rounds is significantly smaller than the diameter of the graph.

The problem of computing a minimum weight spanning tree in this model is a central one. Kutten and Peleg [20] provided an algorithm that works in $O(\sqrt{n}\log^* n + D)$ rounds, where D denotes the diameter of the graph. Their result is nearly optimal in terms of the complexity in n, as shown by Peleg and Rubinovich [34] who provided a lower bound of $\Omega(\sqrt{n}/\log n)$ rounds (when the length of the messages must be bounded).

Another problem studied in the distributed setting that is related to the one studied in this paper, is finding a sparse spanner. The requirement for spanners is much stronger since the distortion of the distance should be as minimal as possible. Thus, to achieve this property, it is usually the case that the number of edges of the spanner is super-linear in n. Pettie [35] was the first to provide a distributed algorithm for finding a low distortion spanner with $O(n)$ edges without requiring messages of unbounded length or $O(D)$ rounds. The number of rounds of his algorithm is $\log^{1+o(1)} n$. Hence, the standard reduction of [33] yields a local algorithm with a trivial linear bound on the query complexity.

1.3.3 Parallel Algorithms

The problems of computing a spanning tree and a minimum weight spanning tree were studied extensively in the parallel computing model (see, e.g., [6], and the references therein). However, these parallel algorithms have time complexity which is at least logarithmic in n and therefore do not yield an efficient algorithm in the local computation model. See [37, 2] for further discussion on the relationship between the ability to construct local computation algorithms and the parallel complexity of a problem.

1.3.4 Local Cluster Algorithms

Local algorithms for graph theoretic problems have also been given for PageRank computations on the web graph [16, 7, 39, 4, 3]. Local graph partitioning algorithms have been presented in [40, 4, 5, 42, 32], which find subsets of vertices whose internal connections are significantly richer than their external connections in time that depends on the size of the cluster that they output. Even when the size of the cluster is guaranteed to be small, it is not obvious how to use these algorithms in the local computation setting where the cluster decompositions must be consistent among queries to all vertices.

1.3.5 Other Related Sublinear-time Approximation Algorithms for Graphs

The problem of estimating the weight of a minimum weight spanning tree in sublinear time was considered by Chazelle, Rubinfeld and Trevisan [11]. They describe an algorithm whose running time depends on the approximation parameter, the average degree and the range of the weights, but does not directly depend on the number of nodes. A question that has been open since that time, even before local computation algorithms were formally defined, is whether it is possible to quickly determine which edges are in the minimum spanning tree. Our lower bound for spanning trees applies to this question.

2 Preliminaries

The graphs we consider have a known degree bound d, and we assume we have query access to their incidence-lists representation. Namely, for any vertex v and index $1 \leq i \leq d$ it is possible to obtain the i^{th} neighbor of v by performing a query to the graph (if v has less than i neighbors, then a special symbol is returned).[2] If the graph is edge-weighted, then the weight of the edge is returned as well. The number of vertices in the graph is n and we assume that each vertex v has an id, $id(v)$, where there is a full order over the ids.

Let $G = (V, E)$ be a graph. We denote the distance between two vertices u and v in G by $d_G(u, v)$. For vertex $v \in V$ and an integer k, let $\Gamma_k(v, G)$ denote the set of vertices at distance at most k from v and let $C_k(v, G)$ denote the subgraph of G induced by $\Gamma_k(v, G)$. Let $n_k(G) \stackrel{\text{def}}{=} \max_{v \in V} |\Gamma_k(v, G)|$. When the graph G is clear from the context, we shall use the shorthand $d(u, v)$, $\Gamma_k(v)$ and $C_k(v)$ for $d_G(u, v)$, $\Gamma_k(v, G)$ and $C_k(v, G)$, respectively.

▶ **Definition 1** (Local Algorithms for sparse spanning graphs)**.** An algorithm $\mathcal{A}$ is a *local sparse spanning graph algorithm* if, given parameters $n \geq 1$, $\epsilon \geq 0$ and $0 \leq \delta < 1$ and given query

[2] Graphs are allowed to have self-loops and multiple edges, but for our problem we may assume that there are no self-loops and multiple-edges (since the answer on a self-loop can always be negative, and the same is true for all but at most one among a set of parallel edges).

access to the incidence-lists representation of a connected graph $G = (V, E)$, the algorithm $\mathcal{A}$ provides query access to a subgraph of G, $G' = (V, E')$ such that the following hold:

1. G' is connected with probability 1.
2. $|E'| \leq (1 + \epsilon) \cdot n$ with probability at least $1 - \delta$, where the probability is taken over the internal coin flips of $\mathcal{A}$.
3. E' is determined by G and the internal randomness of the oracle.

Namely, on input $(u, v) \in E$, $\mathcal{A}$ returns whether $(u, v) \in E'$ and for any sequence of queries, $\mathcal{A}$ answers consistently with the same G'.

An algorithm $\mathcal{A}$ is a *local sparse spanning graph algorithm with respect to a class of graphs* $\mathcal{C}$ if the above conditions hold, provided that the input graph G belongs to $\mathcal{C}$.

We are interested in local algorithms that have small query complexity, namely, that perform few queries to the graph (for each edge they are queried on) and whose running time (per queried edge) is small as well. As for the question of randomness and the implied space complexity of the algorithms, we assume we have a source of (unbounded) public randomness. Under this assumption, our algorithms do not keep a state and a global space is not required. However, if unbounded public randomness is not available, then we note that for our algorithms this is not an issue: One of our algorithms (see Section 5) is actually deterministic, and for the others, the total number of random bits that is actually required (over all possible queries) is upper bounded by the running time of the algorithm, up to a multiplicative factor of $O(\log n)$. In what follows we sometimes describe a global algorithm first, i.e., an algorithm that reads the entire graph and decides the subgraph G'. After that we describe how to locally emulate the global algorithm. Namely on query $e \in E$, we emulate the global algorithm decision on e while performing only a sublinear number of queries.

3 A Lower Bound for General Bounded-Degree Graphs

▶ **Theorem 2.** *Any local sparse spanning graph algorithm has query complexity* $\Omega(\sqrt{n})$. *This result holds for graphs with a constant degree bound* d *and for constant* $0 \leq \epsilon \leq 2d/3$ *and* $0 \leq \delta < 1/3$.

The full proof can be found in the full version of this paper. Here we give the high-level idea. Let V be a set of vertices and let v_0 and v_1 be a pair of distinct vertices in V. In order to prove the lower bound we construct two families of random d-regular graphs over V, $\mathcal{F}^+_{(v_0,v_1)}$ and $\mathcal{F}^-_{(v_0,v_1)}$. $\mathcal{F}^+_{(v_0,v_1)}$ is the family of d-regular graphs, $G = (V, E)$, for which $(v_0, v_1) \in E$. $\mathcal{F}^-_{(v_0,v_1)}$ is the family of d-regular graphs for which $(v_0, v_1) \in E$ and the removal of (v_0, v_1) leaves the graph with two connected components, each of size [3] [4] $n/2$. We prove that given (v_0, v_1), any algorithm that performs at most $\sqrt{n}/c$ queries for some sufficiently large constant $c > 1$ cannot distinguish the case in which the graph is drawn uniformly at random from $\mathcal{F}^+_{(v_0,v_1)}$ from the case in which the graph is drawn uniformly at random from $\mathcal{F}^-_{(v_0,v_1)}$. Essentially, if the number of queries is at most $\sqrt{n}/c$, then with high constant probability, each new query to the graph returns a new random vertex in both families. By "new vertex" we mean a vertex that neither appeared in the query history nor in the answers history. Since the algorithm must answer consistently with a connected graph G', for every

[3] Although a graph that is drawn uniformly from $\mathcal{F}^+_{(v_0,v_1)}$ (or $\mathcal{F}^-_{(v_0,v_1)}$) might be disconnected, this event happens with negligible probability [8]. Hence, the proof of the lower bound remains valid even if we consider $\mathcal{F}^+_{(v_0,v_1)} \cap \mathcal{C}$ and $\mathcal{F}^-_{(v_0,v_1)} \cap \mathcal{C}$ where $\mathcal{C}$ is the family of connected graphs.

[4] Assume the without loss of generality that $n \cdot d$ is even and that $(n \cdot d)/2$ is odd.

graph in the support of $\mathcal{F}^{-}_{(v_0,v_1)}$ it must answer with probability 1 positively on the query (v_0, v_1). But since the distributions on query-answer histories in both cases are very close statistically, this can be shown to imply that there exist graphs for which the algorithm answers positively on a large fraction of the edges.

4 Graphs with High Expansion

In this section we describe an algorithm that gives meaningful results for graphs in which, roughly speaking, the local neighborhood of almost all vertices expands in a similar rate. In particular this includes graphs with high expansion. In fact we only require that the graph expands quickly for small sets: A graph G is an (s, α)-*vertex expander* if for all sets S of size at most s, $N(S)$ is of size at least $\alpha|S|$, where $N(S)$ denotes the set of vertices adjacent to vertices in S that are not in S. Define $h_s(G)$ to be the maximum α such that G is an (s, α)-vertex expander. We shall prove the following theorem.

▶ **Theorem 3.** *Given a graph $G = (V, E)$ with degree bound d, there is a local sparse spanning graph algorithm with query complexity and running time $(d \cdot s)^{\log_{h_s(G)} d}$ where $s = s(n, \epsilon, \delta) \stackrel{\text{def}}{=} \sqrt{2n/\epsilon} \cdot \log(n/\delta)$.*

By Theorem 3, for bounded degree graphs with high expansion we get query and running time complexity nearly $O(n^{1/2})$. In particular, if $h_s(G) = \Omega(d)$ for $s = s(n, \epsilon, \delta)$ then the complexity is $n^{1/2+O(1/\log d)}$. In fact, even for $h_s(g) \geq d^{1/2+1/\log n}$ the complexity is $o(n)$. Recall that in the construction of our lower bound of $\Omega(n^{1/2})$ we construct a pair of families of d-regular random graphs. In both families, the expansion (of small sets) is $\Omega(d)$, implying that for these families the gap between our lower bound and upper bound is at most $n^{O(1/\log d)}$.

Our algorithm, the local Centers' Algorithm (which appears as Algorithm 1), is based on a global algorithm which is presented in Subsection 4.1. The local Centers' Algorithm appears in Subsection 4.2 and it is analyzed in the proof of Theorem 4.

4.1 The Global Algorithm

For a given parameter k the global algorithm first defines a global partition of (part or all of) the graph vertices in the following randomized manner.

1. Select $\ell = \sqrt{\epsilon n/2}$ *centers* uniformly and independently at random from V, and denote them $v_1, \ldots, v_\ell$.
2. Initially, all vertices are *unassigned*.
3. For $i = 0, \ldots, k$, for $j = 1, \ldots, \ell$:
   Let L^i_j denote the vertices in the i^{th} level of the BFS tree of v_j (where $L^0_j = \{v_j\}$). Assign to v_j all vertices in L^i_j that were not yet assigned to any other $v_{j'}$.

Let $S(v_j)$ denote the set of vertices that are assigned to the center v_j. By the above construction, the subgraph induced by $S(v_j)$ is connected.

The subgraph $G' = (V, E')$ is defined as follows.

1. For each center v, let $E'(v)$ denote the edges of a BFS-tree that spans the subgraph induced by $S(v)$ (where the BFS-tree is determined by the order over the ids of the vertices in $S(v)$). For each center v, put in E' all edges in $E'(v)$.
2. For each vertex w that does not belong to any $S(v)$ for a center v, put in E' all edges incident to w.

3. For each pair of centers u and v, let $P(u,v)$ be the shortest path between u and v that has minimum lexicographic order among all shortest paths (as determined by the ids of the vertices on the path). If all vertices on this path belong either to $S(u)$ or to $S(v)$, then add to E' the single edge $(x,y) \in P(u,v)$ such that $x \in S(u)$ and $y \in S(v)$, where we denote this edge by $e(u,v)$.

In what follows we shall prove that G' is connected and that for k that is sufficiently large, G' is sparse with high probability as well. We begin by proving the latter claim. To this end we define a parameter which determines the minimum distance needed for most vertices to see roughly $\sqrt{n}$ vertices. More formally, define $k^C_{\epsilon,\delta}(G)$ to be the minimum distance k ensuring that all but an $\epsilon/(2d)$-fraction of the vertices have at least $s(n,\epsilon,\delta)$ vertices in their k-neighborhood. That is,

$$k^C_{\epsilon,\delta}(G) \stackrel{\text{def}}{=} \min_k \left\{ \left| \{v : \Gamma_k(v) \geq s(n,\epsilon,\delta)\} \right| \geq (1-\epsilon/(2d))\,|V| \right\} . \tag{1}$$

We next establish that for $k \geq k^C_{\epsilon,\delta}(G)$ it holds that $|E'| \leq (1+\epsilon)n$ with probability at least $1-\delta$, over the random choice of centers. Since for $j = 1,\ldots,\ell$ the sets $E'(v_j)$ are disjoint, we have that $\left|\bigcup_{j=1}^{\ell} E'(v_j)\right| < n$. Since there is at most one edge $e(u,v)$ added to E' for each pair of centers u,v and the number of centers is $\ell = \sqrt{\epsilon n/2}$, the total number of these edges in E' is at most $\epsilon n/2$. Finally, Let $T \subseteq V$ denote the subset of the vertices, v, such that $|\Gamma_k(v)| \geq s(n,\epsilon,\delta)$. Since the centers are selected uniformly, independently at random, for each $w \in T$ the probability that no vertex in $\Gamma_k(w)$ is selected to be a center is at most $(1-\log(n/\delta)/\sqrt{\epsilon n/2})^{\sqrt{\epsilon n/2}} < \delta/n$. By taking a union bound over all vertices in T, with probability at least $1-\delta$, every $w \in T$ is assigned to some center v. Since the number of vertices in $V \setminus T$ is at most $\epsilon n/(2d)$ and each contributes at most d edges to E', we get the desired upper bound on $|E'|$.

It remains to establish that G' is connected. To this end it suffices to prove that there is a path in G' between every pair of centers u and v. This suffices because for each vertex w that is assigned to some center v, there is a path between w and v (in the BFS-tree of v), and for each vertex w that is not assigned to any center, all edges incident to w belong to E'. The proof proceeds by induction on $d(u,v)$ and the sum of the ids of u and v as follows. For the base case consider a pair of centers u and v for which $d(u,v) = 1$. In this case, the shortest path $P(u,v)$ consists of a single edge (u,v) where $u \in S(u)$ and $v \in S(v)$, implying that $(u,v) \in E'$. For the induction step, consider a pair of centers u and v for which $d(u,v) > 1$, and assume by induction that the claim holds for every pair of centers (u',v') such that either $d(u',v') < d(u,v)$ or $d(u',v') = d(u,v)$ and $id(u') + id(v') < id(u) + id(v)$. Similarly to base case, if the set of vertices in $P(u,v)$ is contained entirely in $S(u) \cup S(v)$, then u and v are connected by construction. Namely, $P(u,v) = (u, x_1, \ldots, x_t, y_s, \ldots, y_1, v)$ where $x_1,\ldots,x_t \in S(u)$ and $y_1,\ldots,y_s \in v$. The edge (x_t, y_s) was added to E' and there are paths in the BFS-trees of u and v between u and x_t and between v and y_s, respectively. Otherwise, we consider two cases.

1. There exists a vertex x in $P(u,v)$, and a center (different from u and v), y, such that $x \in S(y)$. Note that this must be the case when $d(u,v) \leq 2k+1$. This implies that either $d(x,y) < d(x,v)$ or that $d(x,y) = d(x,v)$ and $id(y) < id(v)$. Hence, either

$$d(u,y) \leq d(u,x) + d(x,y) < d(u,x) + d(x,v) = d(u,v)$$

or $d(u,y) = d(u,v)$ and $id(u) + id(y) < id(u) + id(v)$. In either case we can apply the induction hypothesis to obtain that u and y are connected. A symmetric argument gives us that v and y are connected.

2. Otherwise, all the vertices on the path $P(u,v)$ that do not belong to $S(u) \cup S(v)$ are vertices that are not assigned to any center. Since E' contains all edges incident to such vertices, u and v and connected in this case as well.

4.2 The Local Algorithm

Algorithm 1 (Centers' Algorithm)

For a random choice of $\ell = \sqrt{\epsilon n/2}$ centers, $v_1, \ldots, v_\ell$ in V (which is fixed for all queries), and for a given parameter k, on query (x,y):

1. Perform a BFS to depth k in G from x and from y.
2. If either $\Gamma_k(x) \cap \{v_1, \ldots, v_\ell\} = \emptyset$ or $\Gamma_k(y) \cap \{v_1, \ldots, v_\ell\} = \emptyset$, then return YES.
3. Otherwise, let u be the center closest to x and let v be the center closest to y (if there is more than one such center, break ties according to the order $v_1, \ldots, v_\ell$).
4. If $u = v$ then do the following:
   - If $d(x,u) = d(y,u)$, then return NO.
   - If $d(y,u) = d(x,u) + 1$, then consider all neighbors of y, w, on a shortest path between y and u. If there exists such neighbor w for which $id(w) < id(x)$, then return NO, otherwise, return YES.
5. If $u \neq v$, then perform a BFS of depth k from both of the centers, u and v. Find the shortest path between u and v that has the smallest lexicographical order, and denote it by $P(u,v)$. Return YES if both $x \in P(u,v)$ and $y \in P(u,v)$. Otherwise, return NO.

▶ **Theorem 4.** *Algorithm 1, when run with $k \geq k^C_{\epsilon,\delta}(G)$, is a local sparse spanning graph algorithm. The query complexity and running time of the algorithm are $O(d \cdot n_k(G))$.*

Proof. We prove the theorem by showing that Algorithm 1 is a local emulation of the global algorithm that appears in Subsection 4.1. Given x and y, by performing a BFS to depth k from each of the two vertices, Algorithm 1 either finds the centers u and v that x and y are (respectively) assigned to (by the global algorithm, for the same selection of centers), or for at least one of them it finds no center in the distance k neighborhood. In the latter case, the edge (x,y) belongs to E', and Algorithm 1 returns a positive answer, as required. In the former case, there are two subcases.

1. If x and y are assigned to the same center, that is, $u = v$, then Algorithm 1 checks whether the edge (x,y) is an edge in the BFS-tree of u (i.e., $(x,y) \in E'(u)$). If x and y are on the same level of the tree (i.e., are at the same distance from u), then Algorithm 1 returns a negative answer, as required. If y is one level further than x, then Algorithm 1 checks whether y has another neighbor w that is also assigned to u, is on the same level as x and has a smaller id than x. Namely, a neighbor of y that is on a shortest path between y and u and has a smaller id than x. If this is the case, then the edge (x,y) does not belong to the tree (but rather the edge (w,y)) so that the algorithm returns a negative answer. If no such neighbor of y exists, then the algorithm returns a positive answer (as required).
2. If x and y are assigned to different centers, that is, $u \neq v$, then Algorithm 1 determines whether $(x,y) = e(u,v)$ exactly as defined in the global algorithm: The algorithm finds $P(u,v)$ and returns a positive answer if and only if (x,y) belongs to $P(u,v)$. Notice that from the fact that $x \in S(u)$ and $y \in S(v)$ and the fact that (x,y) belongs to $P(u,v)$ it follows that all the vertices on $P(u,v)$ belong to either $S(u)$ or $S(v)$. This is implied

from the fact that for every center u and a vertex which is assigned to u, w, it holds that every vertex on a shortest path between u and w is also assigned to u.

Finally, the bound on the query complexity and running time of Algorithm 1 follows directly by inspection of the algorithm. ◀

4.3 The Parameter k

Recall that Algorithm 1 is given a parameter k that determines the depth of the BFS that the algorithm performs. By Theorem 4 is suffices to require that $k \geq k^C_{\epsilon,\delta}(G)$ in order to ensure that the spanning graph obtained by the algorithm is sparse. For the case that k is not given in advance we describe next how to compute k such that with probability at least $1-\delta$ it holds that

$$k^C_{\epsilon,\delta}(G) \leq k \leq k'^C_{\epsilon,\delta}(G)\ , \tag{2}$$

where $k'^C_{\epsilon,\delta}(G) = \min_k \left\{ |\{v : \Gamma_k(v) \geq s(n,\epsilon,\delta)\}| \geq \left(1 - \frac{\epsilon d}{4}\right)|V| \right\}$. Select uniformly at random $s = \Theta(1/\epsilon^2 \log(1/\delta))$ vertices from V. Let $v_1, \ldots, v_s$ denote the selected vertices. For each vertex in the sample v_i, let $k_i = \min_k\{\Gamma_k(v_i) \geq s(n,\epsilon,\delta)\}$. Assume without loss of generality that $k_1 \leq \ldots, \leq k_s$ and set $k = k_{\lceil 1 - \frac{3\epsilon}{8d} \rceil}$. By Chernoff's inequality we obtain that with probability greater than $1-\delta$ Equation (2) holds.

We are now ready to prove Theorem 3.

Proof of Theorem 3. Assume without loss of generality that $k^C_{\epsilon,\delta}(G)$ is unknown and we run Algorithm 1 with k such that $k^C_{\epsilon,\delta}(G) \leq k \leq k'^C_{\epsilon,\delta}(G)$. From the fact that $n_k(G) \geq \min\{h_s(G)^k, s\}$ we obtain that $k'^C_{\epsilon,\delta}(G) \leq \frac{\log s}{\log h_s(G)}$ for $s = s(n,\epsilon,\delta)$. On the other hand, since the degree is bounded by d, it holds that $n_k(G) \leq 1 + d^k$. Hence, by Theorem 4 we obtain that the query complexity is bounded by $d \cdot (1 + d^{\frac{\log s}{\log h_s(G)}})$, as desired. ◀

5 Hyperfinite Graphs

In this section we provide an algorithm that is designed for the family of hyperfinite graphs. Roughly speaking, hyperfinite graphs are non-expanding graphs. Formally, a graph $G = (V, E)$ is (ϵ, k)-*hyperfinite* if it is possible to remove at most $\epsilon|V|$ edges of the graph so that the remaining graph has connected components of size at most k. We refer to these edges as the *separating edges* of G. A graph G is ρ-*hyperfinite* for $\rho : \mathbb{R}_+ \to \mathbb{R}_+$ if for every $\epsilon \in (0,1]$, G is $(\epsilon, \rho(\epsilon))$-hyperfinite. The family of hyperfinite graphs includes many subfamilies of graphs such as graphs with an excluded-minor (e.g. planar graphs), graphs that have subexponential growth and graphs with bounded treewidth. The complexity of our algorithm does not depend on the size of the graph as stated in the next theorem.

▶ **Theorem 5.** *Algorithm 2, when run with $k = \rho(\epsilon)$, is a local sparse spanning graph algorithm for the family of ρ-hyperfinite graphs with a degree bounded by d. The query complexity and running time of Algorithm 2 are $O(d^{\rho(\epsilon)+1})$, and its success probability is 1.*

We note that we could also obtain a local sparse spanning graph algorithm for hyperfinite graphs by using the partition oracle of [15] (see the reduction described in Section 6) but the complexity would be higher ($O(d^{d^{\rho(\epsilon)}})$).

We present Algorithm 2 in Subsection 5.1. In Subsection 5.2 we give an improved analysis for the subfamily of graphs that have subexponential growth.

5.1 The Algorithm

Recall that Kruskal's algorithm for finding a minimum-weight spanning tree in a weighted connected graph works as follows. First it sorts the edges of the graph from minimum to maximum weight (breaking ties arbitrarily). Let this order by $e_1, \ldots e_m$. It then goes over the edges in this order, and adds e_i to the spanning tree if and only if it does not close a cycle with the previously selected edges. It is well known (and easy to verify), that if the weights of the edges are distinct, then there is a single minimum weight spanning tree in the graph. For an unweighted graph G, consider the order defined over its edges by the order of the ids of the vertices. Namely, we define a ranking r of the edges as follows: $r(u,v) < r(u',v')$ if and only if $\min\{id(u), id(v)\} < \min\{id(u'), id(v')\}$ or $\min\{id(u), id(v)\} = \min\{id(u'), id(v')\}$ and $\max\{id(u), id(v)\} < \max\{id(u'), id(v')\}$. If we run Kruskal's algorithm using the rank r as the weight function (where there is a single ordering of the edges), then we obtain a spanning tree of G.

While the local algorithm we give in this section (Algorithm 2) is based on the aforementioned global algorithm, it does not exactly emulate it, but rather emulates a certain *relaxed* version of it. In particular, it will answer YES for every edge selected by the global algorithm (ensuring connectivity), but may answer YES also on edges not selected by the global algorithm.

Algorithm 2 (Kruskal-based Algorithm)

The algorithm is provided with an integer parameter k, which is fixed for all queries. On query (x, y):

1. Perform a BFS to depth k from x, thus obtaining the subgraph $C_k(x)$ induced by $\Gamma_k(x)$ in G.
2. If (x, y) is the edge with largest rank on some cycle in $C_k(x)$, then answer NO, otherwise, answer YES.

Theorem 5. By the description of Algorithm 2 it directly follows that its answers are consistent with a connected subgraph G'. We next show that the algorithm returns YES on at most $(1+\epsilon)n$ edges. Let $k = \rho(\epsilon)$. For a vertex u, let $\widetilde{C}(u) = (\widetilde{V}(u), \widetilde{E}(u))$ denote the component of u after the removal of the separating edges (as defined at the start of the subsection). We next prove that G' does not contain a cycle on the subgraph induced on $\widetilde{V}(u)$. In our proof we use properties of $\widetilde{C}(u)$, however, we note that the algorithm does not compute $\widetilde{C}(u)$. By definition, $|\widetilde{V}(u)| \leq k$, thus the diameter of $\widetilde{C}(u)$ is at most $k-1$. This implies that $C_k(u)$ contains $\widetilde{C}(u)$ for every $u \in G$. Let σ be a cycle in $\widetilde{C}(u)$ and let $e = (w, v)$ be the edge in σ with the largest rank. Since $\widetilde{C}(u) = \widetilde{C}(v) = \widetilde{C}(w)$ it follows that on query (w, v) the algorithm returns NO. We conclude that for every $u \in V$ the algorithm returns YES only on at most $|\widetilde{V}(u)| - 1$ among the edges in $\widetilde{E}(u)$. Since the number of edges that do not belong to any component $\widetilde{C}(u)$, that is, the number of separating edges in an $(\epsilon, k = \rho(\epsilon))$-hyperfinite graph is at most $\epsilon|V|$ we have that the total number of edges for which the algorithm returns YES is at most $(1+\epsilon)|V|$. ◀

5.2 Graphs with Subexponential-Growth

In this subsection we analyze Algorithm 2 when executed on graphs with subexponential-growth and for an appropriate k. We first show that graphs with subexponential-growth are $(\epsilon, \rho(\epsilon))$-hyperfinite. In order to obtain an improved analysis of the complexity of Algorithm 2

for graphs with subexponential-growth, we bound not only the size of each component but also the diameter of each component.

A monotone function $f : \mathbb{N} \to \mathbb{N}$ has *subexponential growth* if for any $\beta > 0$, there exists $r_f(\beta) > 0$ such that $f(r) \leq \exp(\beta \cdot r)$ for all $r \geq r_f(\beta)$. A graph G has *growth bounded by* f if for every $k \geq 1$, $n_k(G) \leq f(k)$.

▶ **Theorem 6.** *Given a graph $G = (V, E)$ with degree bounded by d that has growth bounded by $f : \mathbb{N} \to \mathbb{N}$ where f has subexponential growth, there is a local sparse spanning graph algorithm with query complexity and running time $O(d \cdot n_{r_f(\beta)}(G)) = O(d \cdot \exp(\beta \cdot r_f(\beta)))$ for $\beta = \frac{\epsilon}{2d}$.*

Recall that Algorithm 2 is provided with an integer parameter, k, which determines the depth of the BFS that is performed by the algorithm. In case the graph is $(\epsilon, \rho(\epsilon))$-hyperfinite we showed that setting $k = \rho(\epsilon)$ is sufficient. For a general graph G, we next define another parameter which is also sufficient for bounding the required depth of the BFS, as we show in Theorem 7. Thereafter, we shall prove that for graphs with subexponential-growth this parameter is small and can be computed efficiently.

Define $k^K_{\alpha,\beta}(G)$ to be the minimum distance k ensuring that all but an α-fraction of the vertices have at most $\exp(\beta k/2)$ vertices in their k-neighborhood (k is allowed to be larger than the diameter of G so that $k^K_{\alpha,\beta}(G)$ is well defined). Formally,

$$k^K_{\alpha,\beta}(G) = \min_k \left\{ \left| \{ v : |\Gamma_k(v)| \leq \exp(\beta k/2) \} \right| \geq (1-\alpha)\,|V| \right\} \ . \tag{3}$$

▶ **Theorem 7.** *Algorithm 2, when run with $k \geq k^K_{\alpha,\beta}(G)$, where $\alpha + \beta = \epsilon/d$, is a local sparse spanning graph algorithm. The query complexity and running time of the algorithm are $O(dn_k(G)) = O(d^{k+1})$.*

Theorem 6 follows directly from Theorem 7 and Theorem 7 follows directly from the proof of Theorem 5 and the following lemma.

▶ **Lemma 8.** *Every graph $G = (V, E)$ is $(\epsilon, (1+\beta)^k)$-hyperfinite for $k = k^K_{\alpha,\beta}(G)$ and $\alpha + \beta = \epsilon/d$. Moreover, it is possible to remove at most $\epsilon|V|$ edges of the graph so that the remaining graph has connected components with* diameter *at most $2k$.*

Proof. Let $S \subseteq V$ denote the set of vertices, v, for which $|\Gamma_k(v)| > \exp(\beta k/2)$. We start by removing all the edges adjacent to vertices in S. Overall, we remove at most $d\alpha|V|$ edges. For each vertex $v \in V - S$ it holds that $|\Gamma_k(v)| \leq \exp(\beta k/2)$. From the fact that $\exp(x) < 1 + 2x$ for every $x < 1$ we obtain that $|\Gamma_k(v)| < (1+\beta)^k$. Therefore, there exists $k' < k$ such that $|\Gamma_{k'+1}(v)| < |\Gamma_{k'}(v)|(1+\beta)$. Thus, $C_{k'}(v)$ can be disconnected from G by removing at most $d\beta|\Gamma_{k'}(v)|$ edges. Since it holds that $|\Gamma_k(v)| < (1+\beta)^k$ for every subgraph of G and every $v \in V - S$, we can continue to iteratively disconnect connected components of diameter at most $2k$ from the resulting graph. Hence, we obtain that by removing at most $d(\alpha+\beta)|V|$ edges, the remaining graph has connected components with diameter at most $2k$, as desired. ◀

5.3 The Parameter k

Recall that Algorithm 2 is given a parameter k that determines the depth of the BFS that the algorithm performs. By Theorem 7 it is sufficient to require that $k \geq k^K_{\epsilon/(2d),\epsilon/(2d)}(G)$ in order to ensure that the resulting graph is sparse. For the case that k is not given in advance, we can compute k such that with probability greater than $1 - \delta$ it holds that

$$k^K_{\epsilon/(2d),\epsilon/(2d)}(G) \leq k \leq k^K_{\epsilon/(4d),\epsilon/(2d)}(G) \ , \tag{4}$$

as follows. Sample $\Theta(1/\epsilon^2 \log(1/\delta))$ vertices. Start with $k = 1$ and iteratively increase k until for at least $\left(1 - \frac{3\epsilon}{8d}\right)$-fraction of the vertices, v, in the sample it holds that $|\Gamma_k(v)| \leq \exp(\epsilon k/(4d))$. By Chernoff's inequality we obtain that with probability greater than $1 - \delta$ Equation (4) holds.

6 Partition Oracle-based Algorithm

In this section we describe a simple reduction from local algorithm for sparse spanning graph to partition oracle. We begin with a few definitions concerning partition oracles.

▶ **Definition 9.** For $\epsilon \in (0, 1]$, $k \geq 1$ and a graph $G = (V, E)$, we say that a partition $\mathcal{P} = (V_1, \ldots, V_t)$ of V is an (ϵ, k)-*partition* (with respect to G), if the following conditions hold:

1. For every $1 \leq i \leq t$ it holds that $|V_i| \leq k$;
2. For every $1 \leq i \leq t$ the subgraph induced by V_i in G is connected;
3. The total number of edges whose endpoints are in different parts of the partition is at most $\epsilon|V|$ (that is, $|\{(v_i, v_j) \in E : v_i \in V_j, v_j \in V_j, i \neq j\}| \leq \epsilon|V|$).

Let $G = (V, E)$ be a graph and let $\mathcal{P}$ be a partition of V. We denote by $g_{\mathcal{P}}$ the function from $v \in V$ to 2^V (the set of all subsets of V), that on input $v \in V$, returns the subset $V_\ell \in \mathcal{P}$ such that $v \in V_\ell$.

▶ **Definition 10** ([15])**.** An oracle $\mathcal{O}$ is a *partition oracle* if, given query access to the incidence-lists representation of a graph $G = (V, E)$, the oracle $\mathcal{O}$ provides query access to a partition $\mathcal{P} = (V_1, \ldots, V_t)$ of V, where $\mathcal{P}$ is determined by G and the internal randomness of the oracle. Namely, on input $v \in V$, the oracle returns $g_{\mathcal{P}}(v)$ and for any sequence of queries, $\mathcal{O}$ answers consistently with the same $\mathcal{P}$. An oracle $\mathcal{O}$ is an (ϵ, k)-*partition oracle with respect to a class of graphs* $\mathcal{C}$ if the partition $\mathcal{P}$ it answers according to has the following properties.

1. For every $V_\ell \in \mathcal{P}$, $|V_\ell| \leq k$ and the subgraph induced by V_ℓ in G is connected.
2. If G belongs to $\mathcal{C}$, then $|\{(u, v) \in E : g_{\mathcal{P}}(v) \neq g_{\mathcal{P}}(u)\}| \leq \epsilon|V|$ with high constant probability, where the probability is taken over the internal coin flips of $\mathcal{O}$.

By the above definition, if $G \in \mathcal{C}$, then with high constant probability the partition $\mathcal{P}$ is an (ϵ, k)-partition, while if $G \notin \mathcal{C}$ then it is only required that each part of the partition is connected and has size at most k.

▶ **Theorem 11.** *If there exists an (ϵ, k)-partition oracle, $\mathcal{O}$, for the family of graphs $\mathcal{C}$ having query complexity $q(\epsilon, k, d, n)$ and running time $t(\epsilon, k, d, n)$, then there exists a local sparse spanning graph algorithm, $\mathcal{A}$, for the family of graphs $\mathcal{C}$, whose success probability is the same as that of $\mathcal{O}$. The running time of of $\mathcal{A}$ is bounded from above by $t(\epsilon, k, d, n) + O(kd)$ and the query complexity of $\mathcal{A}$ is $q(\epsilon, k, d, n) + O(kd)$.*

Proof. On query (u, v) the algorithm $\mathcal{A}$ proceeds as follows:

1. Query $\mathcal{O}$ on u and v and get $g_{\mathcal{P}}(u)$ and $g_{\mathcal{P}}(v)$, respectively.
2. If $g_{\mathcal{P}}(u) \neq g_{\mathcal{P}}(v)$, return YES.
3. Otherwise, let w denote the vertex in $g_{\mathcal{P}}(u)$ such that $id(w)$ is minimal.
4. Perform a BFS on the subgraph induced on $g_{\mathcal{P}}(u)$, starting from w.
5. If (u, v) belongs to the edges of the above BFS then return YES, otherwise, return NO.

The fact that $\mathcal{A}$ returns YES on at most $(1 + \epsilon)|V|$ edges follows from the fact that $\mathcal{P}$ is a partition can that $|\{(u, v) \in E : g_{\mathcal{P}}(v) \neq g_{\mathcal{P}}(u)\}| \leq \epsilon|V|$. The connectivity follows from the fact that the subgraph induced on V_i is connected for every $V_i \in \mathcal{P}$. The additional term of $O(kd)$ in the time and query complexity is due to the BFS performed on $g_{\mathcal{P}}(u)$. ◀

The following corollaries follow from [14] and [21, 22], respectively.

▶ **Corollary 12.** *There exists a local sparse spanning graph algorithm for the family of graphs with bounded treewidth. This algorithm has high constant success probability and its query complexity and running time are* $\mathrm{poly}(1/\epsilon, d)$.

▶ **Corollary 13.** *There exists a local sparse spanning graph algorithm for the family of graphs with a fixed excluded minor. This algorithm has high constant success probability and its query complexity and running time are* $(d/\epsilon)^{O(\log(1/\epsilon))}$.

References

1 N. Ailon, B. Chazelle, S. Comandur, and D. Liu. Property-preserving data reconstruction. *Algorithmica*, 51(2):160–182, 2008.

2 N. Alon, R. Rubinfeld, S. Vardi, and N. Xie. Space-efficient local computation algorithms. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*, pages 1132–1139, 2012.

3 R. Andersen, C. Borgs, J. Chayes, J. Hopcroft, V. Mirrokni, and S. Teng. Local computation of pagerank contributions. *Internet Mathematics*, 5(1–2):23–45, 2008.

4 R. Andersen, F. Chung, and K. Lang. Local graph partitioning using pagerank vectors. In *FOCS'06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 475–486, 2006.

5 R. Andersen and Y. Peres. Finding sparse cuts locally using evolving sets. In *Proceedings of the Forty-First Annual ACM Symposium on the Theory of Computing*, pages 235–244, 2009.

6 D. A. Bader and G. Cong. A fast, parallel spanning tree algorithm for symmetric multiprocessors (smps). *J. Parallel Distrib. Comput.*, 65(9):994–1006, 2005.

7 P. Berkhin. Bookmark-coloring algorithm for personalized pagerank computing. *Internet Mathematics*, 3(1):41–62, 2006.

8 B. Bollobás. *Random Graphs*. Cambridge University Press, 2001.

9 Z. Brakerski. Local property restoring. Unpublished manuscript, 2008.

10 A. Campagna, A. Guo, and R. Rubinfeld. Local reconstructors and tolerant testers for connectivity and diameter. In *Proceedings of the Seventeenth International Workshop on Randomization and Computation (RANDOM)*, pages 411–424, 2013.

11 B. Chazelle, R. Rubinfeld, and L. Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM Journal on Computing*, 34(6):1370–1379, 2005.

12 B. Chazelle and C. Seshadhri. Online geometric reconstruction. In *Proceedings of the Twenty-Second Annual ACM Symposium on Computation Geometry (SoCG)*, pages 386 – 394, 2006.

13 A. Dutta, R. Levi, D. Ron, and R. Rubinfeld. A simple online competitive adaptation of lempel-ziv compression with efficient random access support. In *Proceedings of the Data Compression Conference (DCC)*, pages 113–122, 2013.

14 A. Edelman, A. Hassidim, H. N. Nguyen, and K. Onak. An efficient partitioning oracle for bounded-treewidth graphs. In *Proceedings of the Fifteenth International Workshop on Randomization and Computation (RANDOM)*, pages 530–541, 2011.

15 A. Hassidim, J. A. Kelner, H. N. Nguyen, and K. Onak. Local graph partitions for approximation and testing. In *Proceedings of the Fiftieth Annual Symposium on Foundations of Computer Science*, pages 22–31, 2009.

16 G. Jeh and J. Widom. Scaling personalized web search. In *Proceedings of the 12th International Conference on World Wide Web*, pages 271–279, 2003.

17 M. Jha and S. Raskhodnikova. Testing and reconstruction of Lipschitz functions with applications to data privacy. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS)*, 2011.

18 S. Kale, Y. Peres, and C. Seshadhri. Noise tolerance of expanders and sublinear expander reconstruction. In *Proceedings of the Forty-Ninth Annual Symposium on Foundations of Computer Science*, pages 719–728, 2008.

19 J. B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the AMS*, 7(1):48–50, 1956.

20 S. Kutten and D. Peleg. Fast distributed construction of small k-dominating sets and applications. *Journal of Algorithms*, 28(1):40–66, 1998.

21 R. Levi and D. Ron. A quasi-polynomial time partition oracle for graphs with an excluded minor. In *40th International Colloquium on Automata, Languages, and Programming (ICALP'13)*, pages 709–720, 2013.

22 R. Levi and D. Ron. A quasi-polynomial time partition oracle for graphs with an excluded minor. *CoRR*, abs/1302.3417, 2013.

23 Reut Levi, Dana Ron, and Ronitt Rubinfeld. Local algorithms for sparse spanning graphs. *CoRR*, abs/1402.3609, 2014.

24 N. Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992.

25 L. Trevisan M. Sudan and S. Vadhan. Pseudorandom generators without the XOR lemma. In *Proceedings of the Thirty-First Annual ACM Symposium on the Theory of Computing*, pages 537–546, 1999.

26 Y. Mansour, A. Rubinstein, S. Vardi, and N. Xie. Converting online algorithms to local computation algorithms. In *Automata, Languages and Programming: Thirty-Ninth International Colloquium (ICALP)*, pages 653–664, 2012.

27 Y. Mansour and S. Vardi. A local computation approximation scheme to maximum matching. In *Proceedings of the Sixteenth International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 260–273, 2013.

28 S. Marko and D. Ron. Distance approximation in bounded-degree and general sparse graphs. *ACM Transactions on Algorithms*, 5(2), 2009. Article number 22.

29 A. Mayer, S. Naor, and L. Stockmeyer. Local computations on static and dynamic graphs. In *Proceedings of the 3rd Israel Symposium on Theory and Computing Systems (ISTCS)*, 1995.

30 M. Naor and L. Stockmeyer. What can be computed locally? *SIAM Journal on Computing*, 24(6):1259–1277, 1995.

31 H. N. Nguyen and K. Onak. Constant-time approximation algorithms via local improvements. In *Proceedings of the Forty-Ninth Annual Symposium on Foundations of Computer Science*, pages 327–336, 2008.

32 L. Orecchia and Z. A. Zhu. Flow-based algorithms for local graph clustering. *CoRR*, abs/1307.2855, 2013.

33 M. Parnas and D. Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theoretical Computer Science*, 381(1-3):183–196, 2007.

34 D. Peleg and V. Rubinovich. A near-tight lower bound on the time complexity of distributed minimum-weight spanning tree construction. *SIAM Journal on Computing*, 30(5):1427–1442, 2000.

35 S. Pettie. Distributed algorithms for ultrasparse spanners and linear size skeletons. *Distributed Computing*, 22(3):147–166, 2010.

36 S. Pettie and V. Ramachandran. Randomized minimum spanning tree algorithms using exponentially fewer random bits. *ACM Transactions on Algorithms*, 4(1), 2008.

37 R. Rubinfeld, G. Tamir, S. Vardi, and N. Xie. Fast local computation algorithms. In *Proceedings of The Second Symposium on Innovations in Computer Science (ICS)*, pages 223–238, 2011.

38 M. E. Saks and C. Seshadhri. Local monotonicity reconstruction. *SIAM Journal on Computing*, 39(7):2897–2926, 2010.

39 T. Sarlos, A. Benczur, K. Csalogany, D. Fogaras, and B. Racz. To randomize or not to randomize: Space optimal summaries for hyperlink analysis. In *Proceedings of the 15th International Conference on WorldWide Web*, pages 297–306, 2006.

40 D. Spielman and S. Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the Thirty-Sixth Annual ACM Symposium on the Theory of Computing*, pages 81–90, 2004.

41 Y. Yoshida, M. Yamamoto, and H. Ito. An improved constant-time approximation algorithm for maximum matchings. In *Proceedings of the Forty-First Annual ACM Symposium on the Theory of Computing*, pages 225–234, 2009.

42 Z. A. Zhu, S. Lattanzi, and V. Mirrokni. A local algorithm for finding well-connected clusters. In *Proceedings of the Thirtieth International Conference on Machine Learning*, 2013.

The Complexity of Ferromagnetic Two-spin Systems with External Fields

Jingcheng Liu[1], Pinyan Lu[2], and Chihao Zhang*[1]

1 **Shanghai Jiao Tong University**
800 Dongchuan Road, Shanghai, China
liuexp@gmail.com, chihao.zhang@gmail.com

2 **Microsoft Research**
999 Zixing Road, Shanghai, China
pinyanl@microsoft.com

Abstract

We study the approximability of computing the partition function for ferromagnetic two-state spin systems. The remarkable algorithm by Jerrum and Sinclair showed that there is a fully polynomial-time randomized approximation scheme (FPRAS) for the special ferromagnetic Ising model with any given uniform external field. Later, Goldberg and Jerrum proved that it is #BIS-hard for Ising model if we allow inconsistent external fields on different nodes. In contrast to these two results, we prove that for any ferromagnetic two-state spin systems except the Ising model, there exists a threshold for external fields beyond which the problem is #BIS-hard, even if the external field is uniform.

1998 ACM Subject Classification F.2.2 Computations on discrete structures

Keywords and phrases Spin System, #BIS-hard, FPRAS

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.843

1 Introduction

Spin systems are well studied in statistical physics and applied probability. We focus on two-state spin systems in this paper. An instance of a spin system is described by a graph $G(V, E)$, where vertices are particles and edges indicate neighborhood relation among them. A configuration $\sigma : V \to \{0, 1\}$ assigns one of the two states to every vertex. The contribution of local interactions between adjacent vertices is quantified by a matrix $\mathbf{A} = \begin{bmatrix} A_{0,0} & A_{0,1} \\ A_{1,0} & A_{1,1} \end{bmatrix} = \begin{bmatrix} \beta & 1 \\ 1 & \gamma \end{bmatrix}$, where $\beta, \gamma \geq 0$. The contribution of vertices in different spin states is quantified by a vector $\mathbf{b} = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix} = \begin{bmatrix} \mu \\ 1 \end{bmatrix}$, where $\mu > 0$. This μ is also called the external field of the system, which indicates a priori preference of an isolate vertex. The *partition function* $Z_{(\beta,\gamma,\mu)}(G)$ of a spin system $G(V, E)$ is defined to be the following exponential summation:

$$Z_{(\beta,\gamma,\mu)}(G) \triangleq \sum_{\sigma \in \{0,1\}^V} \prod_{v \in V} b_{\sigma_v} \prod_{(u,v) \in E} A_{\sigma_u, \sigma_v}.$$

We call such a spin system parameterized by (β, γ, μ). If the parameters are clear from the context, we shall write $Z(G)$ for short. Although originated from statistical physics, the

* The author is supported by NSF of China (61033002, ANR 61261130589).

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 843–856

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

spin model is also accepted in computer science as a framework for counting problems. For example, with $\beta = 0$, $\gamma = 1$ and $\mu = 1$, $Z_{(\beta,\gamma,\mu)}(G)$ is the number of independent sets (or vertex covers) of the graph G.

Given a set of parameters (β, γ, μ), it is a computational problem to compute the partition function $Z_{(\beta,\gamma,\mu)}(G)$ with input G. We denote this computation problem as $\text{Spin}(\beta, \gamma, \mu)$ and want to characterize its computational complexity in terms of β, γ and μ. For *exact* computation, polynomial time algorithms only exist for the very restricted settings that $\beta\gamma = 1$ and $(\beta, \gamma) = (0, 0)$. For all other settings, the problem is known to be #P-hard [2]. Therefore, the main focus becomes to study its approximability. The notion of the fully polynomial-time approximation scheme (FPTAS) is defined as follows: A algorithm $\mathcal{A}$ is an FPTAS for $\text{Spin}(\beta, \gamma, \mu)$ if for any given parameter $\varepsilon > 0$, $\mathcal{A}$ outputs a number $\hat{Z}$ such that $Z(G)\exp(-\varepsilon) \leq \hat{Z} \leq Z(G)\exp(\varepsilon)$ and runs in time $poly(n, 1/\varepsilon)$, where n is the size of the graph G. The randomized relaxation of FPTAS is called fully polynomial-time randomized approximation scheme (FPRAS), which uses random bits and only requires the final output be within the required accuracy with high probability.

The spin systems (β, γ, μ) are classified into two families with distinct physical and computational properties: *ferromagnetic* systems ($\beta\gamma > 1$) and *anti-ferromagnetic* systems ($\beta\gamma < 1$). We shall denote the corresponding computation problems by $\text{Ferro}(\beta, \gamma, \mu)$ and $\text{Anti-Ferro}(\beta, \gamma, \mu)$ respectively, so as to emphasize which family these parameters belong to. Systems with $\beta\gamma = 1$ are degenerate and trivial both physically and computationally. As a result, we only study systems with $\beta\gamma \neq 1$.

Great progress has been made recently for approximately computing the partition function for anti-ferromagnetic two-spin systems: it admits an FPTAS up to the uniqueness threshold [20, 12, 13, 16], and is NP-hard to approximate in the non-uniqueness range [18, 6]. The uniqueness threshold is a phase transition boundary in physics. It is widely conjectured that the computational difficulty is related to the phase transition point in many problems; this is one of the very few examples where a rigorous proof is obtained.

For ferromagnetic systems, the picture is quite different. The uniqueness condition does not coincide with the transition of computational difficulty and it is not clear whether they have any relation. In a seminal paper [10], Jerrum and Sinclair gave an FPRAS for ferromagnetic Ising model $\beta = \gamma > 1$ with any external field μ. Thus, there is no transition of computational difficulty for ferromagnetic Ising model, which contrasts the situation for anti-ferromagnetic Ising model $\beta = \gamma < 1$. For general ferromagnetic spin systems with external field, the approximability is less clear. Since the Ising model ($\beta = \gamma$) is solved, in this paper, we focus on the case $\beta \neq \gamma$ and always assume $\beta < \gamma$ by symmetry. It is known that, an FPRAS exists for $\mu \leq \sqrt{\gamma/\beta}$ [9], by a reduction to Ising model.

On the other hand, a hardness result was obtained for Ising model with inconsistent external fields [7]. This is a generalization of the spin system where the external fields for vertices are no longer required to be uniform and are arbitrarily taken from a set $\mathcal{V}$. We use $\text{Spin}(\beta, \gamma, \mathcal{V})$ ($\text{Ferro}(\beta, \gamma, \mathcal{V})$ or $\text{Anti-Ferro}(\beta, \gamma, \mathcal{V})$) to denote this computation problem. It is proved that the Ising model with arbitrary external fields $\text{Ferro}(\beta, \beta, (0, +\infty))$ is #BIS-hard, namely the problem is at least as hard as counting independent sets on bipartite graphs (#BIS). #BIS is a problem of intermediate hardness and has been conjectured to admit no FPRAS [5]. The reduction used here is called approximation-preserving reduction as introduced in [4]: Let $A, B : \Sigma^* \to \mathbb{R}$ be two functions. An *approximation-preserving reduction* from A to B is a randomized polynomial-time algorithm that approximates A while using an oracle for B. We write $A \leq_{AP} B$ for short if an approximation-preserving reduction exists from A to B. To make use of the aforementioned #BIS-hardness result, one needs to

use (or simulate) both arbitrarily small and large external fields. As $\beta < \gamma$, one can always simulate a arbitrarily small external field with a gadget. However, simulating a arbitrarily large external field is only possible when $\beta\mu + 1 > \mu + \gamma$, in which case a #BIS-hardness is immediate. If this is not the case, and in particular if $\beta \leq 1 < \gamma$, no hardness result was known for any bounded external fields. These systems have certain monotonicity property, so all external fields that can be simulated by gadgets are inherently bounded above. It was not even clear whether problems in this regime is hard. As our first result, we show that the problem is already hard as long as we allow sufficiently large (yet still bounded above) and vertex-dependent external fields.

▶ **Theorem 1.** *For any $\beta < \gamma$ with $\beta\gamma > 1$, there exists a bounded set $\mathcal{V}$ such that* Ferro$(\beta, \gamma, \mathcal{V})$ *is* #BIS*-hard.*

The main difficulty to establish the theorem is for the case of $\beta \leq 1$, for which we cannot simulate any external field larger than the upper bound of $\mathcal{V}$. We overcome this difficulty by making use of a recent beautiful result in [3]. Instead of starting with the independent set problem on arbitrary bipartite graphs, we start with a soft ($\beta\gamma > 0$) *anti*-ferromagnetic two-spin system on bipartite graphs of *bounded degree*. As a result, all the external fields needed for the reduction are bounded.

However, in the above reduction, we do need vertices to have different external fields to make the reduction go through. This gives a hardness result for Ferro$(\beta, \gamma, \mathcal{V})$ but not for Ferro(β, γ, μ) for a single μ. It is more interesting and intriguing (both physically and computationally) to understand the computational complexity of a uniform spin system (β, γ, μ) with the same external field μ on all vertices. As our main result of this paper, we prove #BIS-hardness for this uniform case for sufficiently large single external field μ. We prove that when μ is sufficiently large, we can realize by sufficient precision of all the external fields which is smaller than $\mu^*(\mu, \beta, \gamma)$, where $\mu^*(\mu, \beta, \gamma) < \mu$ is a function of μ, β and γ, and approaches infinity as μ goes to infinity. Then by choosing large enough μ and making use of Theorem 1, we obtain our main theorem.

▶ **Theorem 2.** *For any $\beta < \gamma$ with $\beta\gamma > 1$, there exist a μ_0 such that* Ferro(β, γ, μ) *is* #BIS*-hard for all $\mu \geq \mu_0$.*

Our main technical contribution is the construction of a family of gadgets to simulate a given target external field. We use a reverse idea of correlation decay to achieve this. Correlation decay is proved to be a very powerful technique to design FPTAS for counting problems (see for examples [20, 1, 13, 16, 14, 15]). In those correlation decay based FPTASes, one first constructs a tree structure and hopes to compute the marginal probability of the root. With a recursive relation, one writes the marginal probability of the root as a function of that of its sub-trees, then truncates the computation tree at certain depth and applies a rough guess at the leaf nodes. The correlation decay property ensures that the error for the root is exponentially small with respect to the depth of the tree, although there might be constant error for the leaves. To establish Theorem 2, we use a similar idea to construct a tree gadget so that the marginal probability (effective external field) for the root is very close to the target value. Using a tree recursion, one translates the target marginal probability for the root to that of its sub-trees. In the leaf nodes, we simply use some basic gadgets to approximate the target external field. Again, although these approximations for leaves may have constant gaps, the error at the root is exponentially small thanks to the correlation decay property. We believe that this idea of using an algorithm design technique to construct gadgets to establish hardness result is of independent interest and may find applications in other problems.

We also make some improvements on the algorithm side showing that there is an FPRAS if $\mu \leq \gamma/\beta$. We remark that all the computational problem $\text{FERRO}(\beta, \gamma, \mu)$ and $\text{FERRO}(\beta, \gamma, \mathcal{V})$ is no more difficult than #BIS, as we can use the standard transformation to transform any ferromagnetic two-spin system to ferromagnetic Ising model with possibly different external fields and use the #BIS-easiness result in [7]. Thus, the two #BIS-hardness theorems can also be stated as #BIS-equivalent. We believe that the conjecture here is that for any fixed $\beta < \gamma$, there exists a critical μ_c such that it admits an FPRAS if the external field $\mu < \mu_c$, and it is #BIS-equivalent if $\mu > \mu_c$. The result of this paper is an important step towards this dichotomy.

Related Works and Organization of the Paper

The approximation for partition function of spin system and other similar models has been studied extensively [1, 19, 8, 11]. For the anti-ferromagnetic two-state spin model, the problem is known to be tractable up to the uniqueness threshold [16, 13, 18, 6], this includes the hard-core model as a special case [20, 17]. For the ferromagnetic two-state spin model, FPRAS was known for Ising model with arbitrary external field [10] and this was later extended to the whole ferromagnetic regime [9]. Besides the FPRASes, there is also a recent deterministic FPTAS for certain range of the parameters based on correlation decay and holographic reduction [15].

The remainder of the paper is organized as follows. We apply a reduction from a recent established hardness in [3] to prove Theorem 1 in Section 2. Based on this, we construct gadgets that can realize sufficiently small external field and prove Theorem 2 in Section 3. Finally, we present our improved tractable result in Section 4.

2 Bounded Local Fields

In this section, we show that spin systems with bounded local fields are already hard. The following theorem is a formal statement of Theorem 1.

▶ **Theorem 3.** *Let* $\beta < \gamma$, $\beta\gamma > 1$, $\Delta = \lfloor \frac{2\sqrt{\beta\gamma}}{\sqrt{\beta\gamma}-1} \rfloor + 1$ *and* $\mu > \left(\sqrt{\frac{\gamma}{\beta}}\right)^{\Delta}$.
Then $\text{FERRO}(\beta, \gamma, [1, \mu])$ *is* #BIS*-hard.*[1]

We first introduce our starting point from anti-ferromagnetic Ising model on bipartite graphs in Section 2.1, and show the reduction in Section 2.2.

2.1 Anti-ferromagnetic Spin Systems on Bipartite Graphs

#BIS is a special anti-ferromagnetic two-state spin system. Similar to #BIS, one can also study other anti-ferromagnetic two-state spin systems on bipartite graphs. We use a prefix BI- to emphasize that input graphs are bipartite, and a subscript Δ to indicate that maximum degree is Δ. For instance, the problem of $\text{ANTI-FERRO}(\beta, \gamma, \mu)$ on bipartite graphs with maximum degree Δ is denoted shortly by $\text{BI-ANTI-FERRO}_{\Delta}(\beta, \gamma, \mu)$. The following theorem from [3] is the starting point of our reduction.

[1] Technically, we should only define the problem by a finite set of external fields. In this paper and as in many others, we adopt the following convention: when we say a problem with an infinite set of external fields is hard, it means that there exists a finite subset of external fields to make the problem hard already.

▶ **Theorem 4** ([3]). *Suppose a set of anti-ferromagnetic parameters (β,γ,μ) lies in the non-uniqueness region of the infinite Δ-regular tree $\mathbb{T}_\Delta$. Then* Bi-Anti-Ferro$_\Delta(\beta,\gamma,\mu)$ *is* #BIS*-hard except for the case $(\beta=\gamma,\lambda=1)$.*

For simplicity, we use the special anti-ferromagnetic Ising model $\beta=\gamma<1$ in our reduction, for which the non-uniqueness condition is easy to state.

▶ **Proposition 5.** *If $\beta<\frac{\Delta-2}{\Delta}$, then there is a critical activity $\mu_c(\beta,\Delta)>1$ such that the Gibbs measure of Ising model (β,β,μ) on infinite Δ-regular tree $\mathbb{T}_\Delta$ is unique if and only if $|\log\mu|\geq\log\mu_c(\beta,\Delta)$.*

Proposition 5 is folklore, a proof can be found in, e. g. [16]. Combining the two results we get

▶ **Corollary 6.** *For all $0<\beta<1$, there is $\varepsilon>0$ such that for any $\mu\in(1,1+\varepsilon)$,* Bi-Anti-Ferro$_\Delta(\beta,\beta,\mu)$ *is* #BIS*-hard, where $\Delta=\lfloor\frac{2}{1-\beta}\rfloor+1$.*

Proof. As $\Delta=\lfloor\frac{2}{1-\beta}\rfloor+1$, we know that $\beta<\frac{\Delta-2}{\Delta}$. Then by Proposition 5, we can choose $\varepsilon=\mu_c(\beta,\Delta)-1$ to ensure that (β,β,μ) is in the non-uniqueness region of the infinite Δ-regular tree $\mathbb{T}_\Delta$ for all $\mu\in(1,1+\varepsilon)$. Then the corollary follows from Theorem 4. ◀

2.2 The Reduction

▶ **Lemma 7.** *For any $\beta<\gamma$ with $\beta\gamma>1$, $\mu>1$ and integer $\Delta>1$, we have*

$$\text{Bi-Anti-Ferro}_\Delta\left(\frac{1}{\sqrt{\beta\gamma}},\frac{1}{\sqrt{\beta\gamma}},\mu\right)\leq_{AP}\text{Bi-Ferro}_\Delta\left(\beta,\gamma,\left[\frac{1}{\mu}\sqrt{\frac{\gamma}{\beta}},\mu\left(\sqrt{\frac{\gamma}{\beta}}\right)^\Delta\right]\right).$$

Proof. Let bipartite graph $G(L\cup R,E)$ be an instance of Bi-Anti-Ferro$_\Delta\left(\frac{1}{\sqrt{\beta\gamma}},\frac{1}{\sqrt{\beta\gamma}},\mu\right)$. We construct an instance of ferromagnetic system with exactly the same graph. Each vertex $u\in L$ with degree d_u has weight $\mu\left(\sqrt{\frac{\gamma}{\beta}}\right)^{d_u}$, and each vertex $v\in R$ has weight $\frac{1}{\mu}\left(\sqrt{\frac{\gamma}{\beta}}\right)^{d_v}$. Then the maximum possible external field is $\mu\left(\sqrt{\frac{\gamma}{\beta}}\right)^\Delta$ while the minimum one is $\frac{1}{\mu}\sqrt{\frac{\gamma}{\beta}}$. Therefore, it is indeed an instance of Bi-Ferro$_\Delta\left(\beta,\gamma,\left[\frac{1}{\mu}\sqrt{\frac{\gamma}{\beta}},\mu\left(\sqrt{\frac{\gamma}{\beta}}\right)^\Delta\right]\right)$.

Let $Z_1(G)$ be the partition function of the anti-ferromagnetic Ising instance, and $Z_2(G)$ be that for the ferromagnetic system. We shall prove that $Z_1(G)=\gamma^{-|F|}\mu^{|R|}Z_2(G)$. Let $V\triangleq L\cup R$, $A=\begin{bmatrix}\frac{1}{\sqrt{\beta\gamma}}&1\\1&\frac{1}{\sqrt{\beta\gamma}}\end{bmatrix}$, $A'=\begin{bmatrix}\sqrt{\frac{\gamma}{\beta}}&\gamma\\\gamma&\sqrt{\frac{\gamma}{\beta}}\end{bmatrix}$, $\hat{A}'=\begin{bmatrix}1&\beta\\\gamma&1\end{bmatrix}$ and $\hat{A}=\begin{bmatrix}\beta&1\\1&\gamma\end{bmatrix}$. Then

$$\begin{aligned}
Z_2(G)&=\sum_{\sigma\in\{0,1\}^V}\prod_{(u,v)\in E}\hat{A}_{\sigma_u,\sigma_v}\prod_{u\in L}\left(\mu\left(\sqrt{\frac{\gamma}{\beta}}\right)^{d_u}\right)^{1-\sigma_u}\prod_{v\in R}\left(\frac{1}{\mu}\left(\sqrt{\frac{\gamma}{\beta}}\right)^{d_v}\right)^{1-\sigma_v}\\
&=\sum_{\sigma\in\{0,1\}^V}\prod_{(u,v)\in E}\hat{A}'_{\sigma_u,\sigma_v}\prod_{u\in L}\left(\mu\left(\sqrt{\frac{\gamma}{\beta}}\right)^{d_u}\right)^{1-\sigma_u}\prod_{v\in R}\left(\frac{1}{\mu}\left(\sqrt{\frac{\gamma}{\beta}}\right)^{d_v}\right)^{\sigma_v}\\
&=\sum_{\sigma\in\{0,1\}^V}\prod_{(u,v)\in E}A'_{\sigma_u,\sigma_v}\prod_{u\in L}\mu^{1-\sigma_u}\prod_{v\in R}\frac{1}{\mu^{\sigma_v}}\\
&=\mu^{-|R|}\gamma^{|F|}\sum_{\sigma\in\{0,1\}^V}\prod_{(u,v)\in E}A_{\sigma_u,\sigma_v}\prod_{u\in L}\mu^{1-\sigma_u}\prod_{v\in R}\mu^{1-\sigma_v}\\
&=\mu^{-|R|}\gamma^{|F|}Z_1(G).
\end{aligned}$$

Thus we can get an approximation for the anti-ferromagnetic Ising model by an oracle call to the ferromagnetic two-spin system. This concludes the proof. ◀

Now, for the target $\mu > \left(\sqrt{\frac{\gamma}{\beta}}\right)^{\Delta}$ in Theorem 3, we simply choose μ' close enough to 1 in Lemma 7 and Corollary 6, such that $\left[\frac{1}{\mu'}\sqrt{\frac{\gamma}{\beta}}, \mu'\left(\sqrt{\frac{\gamma}{\beta}}\right)^{\Delta}\right] \subseteq [1,\mu]$ and $\#\text{BIS} \leq_{AP}$ Bi-Anti-Ferro$_{\Delta}\left(\frac{1}{\sqrt{\beta\gamma}}, \frac{1}{\sqrt{\beta\gamma}}, \mu'\right)$. Then we can conclude that Bi-Ferro$_{\Delta}(\beta,\gamma,[1,\mu])$ is #BIS-hard and complete the proof of Theorem 3.

3 Uniform Local Field

We establish Theorem 2 in this section. We distinguish between $\beta \leq 1$ and $\beta > 1$ cases, in Section 3.1 and 3.2 respectively.

3.1 The $\beta \leq 1$ case

We introduce a function $h(x) = \frac{\beta x+1}{x+\gamma}$ which is used throughout this section. Note that since $\beta\gamma > 1$, $h(x)$ is monotonically increasing and $\frac{1}{\gamma} < h(x) < \beta \leq 1$ for $x \in (0,+\infty)$. We shall prove the following key reduction.

▶ **Lemma 8.** *Let $\beta \leq 1, \beta\gamma > 1$, d be an integer such that $\beta(\beta\gamma)^d > 1$, μ^* be the largest solution of x to $x = \mu h(x)^d$, and $\mu > \frac{\gamma^d(\beta\gamma-1)}{\beta}\left(1+\frac{d+1}{\ln(\beta(\beta\gamma)^d)}\right)$. Then* Ferro $(\beta,\gamma,[1,\mu^*]) \leq_{AP}$ Ferro(β,γ,μ).

As $\mu^* = \mu h(\mu^*)^d$ and $\frac{1}{\gamma} < h(\mu^*) < \beta$, we have the following bound for μ^*.

▶ **Proposition 9.** $\frac{\mu}{\gamma^d} < \mu^* < \beta^d\mu$.

With this bound and Lemma 8, we can choose sufficiently large μ so that this μ^* is large enough to apply the hardness result (Theorem 3) of Ferro$(\beta,\gamma,[1,\mu^*])$ to get the hardness result for Ferro(β,γ,μ). Formally, we have

▶ **Theorem 10.** *Let $\beta \leq 1, \beta\gamma > 1$, d be an integer such that $\beta(\beta\gamma)^d > 1$, $\Delta = \lfloor\frac{2\sqrt{\beta\gamma}}{\sqrt{\beta\gamma}-1}\rfloor + 1$, and $\mu > \gamma^d \max\left\{\left(\sqrt{\frac{\gamma}{\beta}}\right)^{\Delta}, \frac{\beta\gamma-1}{\beta}\left(1+\frac{d+1}{\ln(\beta(\beta\gamma)^d)}\right)\right\}$. Then* Ferro$(\beta,\gamma,\mu)$ *is* #BIS*-hard.*

We remark that there always exists such integer d since $\beta > 0$ and $\beta\gamma > 1$. Different ds give different bounds for μ and it is not necessarily monotone. For a given β,γ, one can choose a suitable d to get the best bound[2].

In the remaining of this section, we prove the key reduction stated in Lemma 8. The main idea is to simulate any external field in $[1,\mu^*]$ by a vertex weight gadget. In Section 3.1.1 , we state the general framework of such simulation. Then in Section 3.1.2, we present the detailed construction of a gadget.

[2] We give one numerical example here to get some idea of this bound: if $\beta = 1$ and $\gamma = 2$, we can get $\Delta = 7$ and choose $d = 1$; then the theorem tells us that the problem Ferro$(1,2,\mu)$ is #BIS-hard if $\mu > 16\sqrt{2}$.

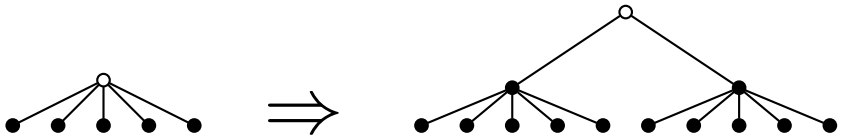


Figure 1 Result of $\mathcal{S}_5 \Rightarrow \texttt{comb}(\{\mathcal{S}_5, \mathcal{S}_5\})$, the output vertex is marked as unfilled.

3.1.1 Vertex Weight Gadget

▶ **Definition 11** (Vertex weight gadget)**.** Let $G(V, E)$ be a graph with a special output vertex v^*, define $\mu(G) = \frac{Z_G(v^*=0)}{Z_G(v^*=1)}$ where $Z_G(v^* = 0)$ (resp. $Z_G(v^* = 1)$) is the partition function of $G(V, E)$ in (β, γ, μ)-system conditioned on $v^* = 0$ (resp. $v^* = 1$). We call G a vertex weight gadget that realizes $\mu(G)$.

We also use a family of graphs to approach a given external field. Let $\{G_i\}_{i\geq 1}$ be a family of vertex weight gadgets. We say $\{G_i\}$ realizes μ if $\lim_{i\to\infty} \mu(G_i) = \mu$.

Vertex weight gadgets can be used to simulate external fields. Formally, we have the following reductions.

▶ **Lemma 12.** *Let G be a vertex weight gadget of $(\beta, \gamma, \mathcal{V})$. Then* $\textsc{Spin}(\beta, \gamma, \mathcal{V} \cup \{\mu(G)\}) \leq_{AP} \textsc{Spin}(\beta, \gamma, \mathcal{V})$.

Let $\{G_i\}$ be a sequence of vertex weight gadget of $(\beta, \gamma, \mathcal{V})$ to realize μ such that for any $\varepsilon > 0$ there is a G_i of size poly $\left(\varepsilon^{-1}\right)$ with $\exp(-\varepsilon) \leq \frac{\mu(G_i)}{\mu} \leq \exp(\varepsilon)$. Then $\textsc{Spin}(\beta, \gamma, \mathcal{V} \cup \{\mu\}) \leq_{AP} \textsc{Spin}(\beta, \gamma, \mathcal{V})$.

Proof. The proof of the first part is straightforward. For any instance H of $\textsc{Spin}(\beta, \gamma, \mathcal{V} \cup \{\mu(G)\})$ and a vertex of H with external field $\mu(G)$, we use one copy of G and identify the output vertex of G with that chosen vertex of H. After the identification, the external field in that vertex is that of output vertex of G. Therefore, after the modification, the new instance is an instance of $\textsc{Spin}(\beta, \gamma, \mathcal{V})$ and the partition function is equal to the partition function of H scaled by a polynomial-time computable global factor $\left(\frac{Z(G)}{1+\mu(G)}\right)^j$, where j is the number of vertices with external field $\mu(G)$ in H.

For the second part, for an instance H of $\textsc{Spin}(\beta, \gamma, \mathcal{V} \cup \{\mu\})$ and required approximation parameter ε, choose a gadget G_i which is $\varepsilon' = \frac{\varepsilon}{2n}$ close to realize μ; do the same modification as above using this G_i and call the oracle for the new instance with approximation parameter ε'. This gives the desired approximation for the original instance. ◀

3.1.2 The Construction

We first define a gadget operation comb as follows: for a given list of graphs $\mathcal{G} = \{G_1, \ldots, G_k\}$, each with output v_i^* for $i \in [k]$, $\texttt{comb}(\mathcal{G})$ is a new graph $G(V, E)$ that combines the graphs and joins their outputs. Fig. 1 is an illustration of comb. Formally, we define $V = \{u\} \cup \bigcup_{i\in[k]} V(G_i)$ and $E = \{(u, v_i^*) \mid i \in [k]\} \cup \bigcup_{i\in[k]} E(G_i)$, where u is the output of G. It is easy to verify that $\mu(G) = \mu \prod_{i\in[k]} h(\mu(G_i))$.

We also define two basic gadgets. Let $\mathcal{S}_w$ be a w-star graph, with output being its center. In particular, $\mathcal{S}_0$ is the singleton graph. Note that $\mu(\mathcal{S}_w) = \mu h(\mu)^w$. We also define $\mathcal{T}_t$ be a d-ary tree with depth t. For any external field $\hat{\mu} \in (0, \mu^*]$, we shall construct a list of gadgets to simulate it. The two boundaries are approached by $\mathcal{S}_w$ and $\mathcal{T}_t$ respectively.

▶ **Proposition 13.** *Let $\mathcal{S}_w$ be a w-star and $\mathcal{T}_t$ be a d-ary tree with depth t. Then*
1. *$\{\mathcal{S}_w\}_{w\geq 1}$ realizes 0, or formally, $\mu(\mathcal{S}_w) = \mu h(\mu)^w < \mu\beta^w$.*
2. *$\{\mathcal{T}_t\}_{t\geq 0}$ realizes μ^*, or formally, there exist two positive constants ι and $c < 1$ depending on μ, β, γ and d such that $1 < \frac{\mu(\mathcal{T}_t)}{\mu^*} \leq \exp(c^t\iota)$.*

Proof. (1) is obvious, we only prove (2).

Note that $\mu(\mathcal{T}_t) = \mu h(\mu(\mathcal{T}_{t-1}))^d$, we denote $f(x) = \mu h(x)^d$. Recall that μ^* is the largest fixed point of $f(x)$ and $f(\mu) < \mu$, we have $0 < f'(\mu^*) < 1$. Define $g(x) = \frac{xf'(x)}{f(x)}$, then $g(\mu^*) = f'(\mu^*)$. Since $g(x)$ is a continuous function, we can choose some $\eta > 0$ such that $0 < g(x) \leq c < 1$ for all $x \in (\mu^* - \eta, \mu^* + \eta)$.

We now define a sequence $\{x_i\}_{i\geq 0}$ such that $x_0 = \mu$ and $x_i = f(x_{i-1})$ for all $i \geq 1$. We claim that $\{x_i\}$ converges to μ^* as i approaches infinity. To see this, note that $x_{i+1} = f(x_i) < x_i$ and $x_i > \mu^*$ for all $i \geq 0$. This implies $\{x_i\}$ converges to some $z \geq \mu^*$. Moreover, since f is continuous, the sequence $\{f(x_i)\}_{i\geq 0}$ also converges to z. These two facts together imply $z = \lim_{i\to\infty} f(x_i) = f(\lim_{i\to\infty} x_i) = f(z)$. In other word, z is a fixed point of f and thus $z = \mu^*$. The claim implies that for some integer t_0, $x_{t_0} \in (\mu^*, \mu^* + \eta)$.

We define another sequence $\{y_i\}_{i\geq 0}$ such that $y_0 = \mu(\mathcal{T}_{t_0})$ and $y_i = f(y_{i-1})$ for all $i \geq 1$. It holds that $y_i \in (\mu^*, \mu^* + \eta)$ and thus $g(y_i) \leq c < 1$ for all $i \geq 0$. Therefore for all $t \geq 1$,

$$\begin{aligned}
\ln y_t - \ln \mu^* &= \ln f(y_{t-1}) - \ln f(\mu^*) \\
&= \frac{\tilde{y}f'(\tilde{y})}{f(\tilde{y})} \cdot |\ln y_{t-1} - \ln \mu^*| \quad \text{for some } y \in [\mu^*, y_{t-1}] \\
&= g(\tilde{y}) \cdot |\ln y_{t-1} - \ln \mu^*| \\
&\leq c \cdot |\ln y_{t-1} - \ln \mu^*| \\
&\leq c^t \eta.
\end{aligned}$$

We denote $\iota = \max\{\ln \mu, \eta c^{-t_0}\}$ and conclude the proof. ◀

Let d be the one that satisfies the requirement in the statement of Lemma 8. Our main idea to realize a target external field $\hat{\mu}$ is to construct a list of gadgets $\mathcal{G} = \{G_1, \ldots, G_d\}$ such that $\mu(\mathtt{comb}(\mathcal{G})) \approx \hat{\mu}$ or more concretely $\hat{\mu} \approx \mu \prod_{i\in[d]} h(\mu(G_i))$. All but one of these G_i are basic gadgets of the following three types:

1. isolate point $\mathcal{S}_0$ with $\mu(\mathcal{S}_0) = \mu$;
2. $\mathcal{S}_w$ with large enough w such that $\mu(\mathcal{S}_w) \approx 0$; and
3. $\mathcal{T}_t$ with large enough t such that $\mu(\mathcal{T}_t) \approx \mu^*$.

The remaining one G_i is recursively constructed with a new target $\hat{\mu}'$ so that ideally $\hat{\mu} = \mu \prod_{i\in[d]} h(\mu(G_i))$ holds. The combination of these basic gadgets are carefully chosen so that the new target $\hat{\mu}'$ is also in the range $(0, \mu^*]$. Then we recursively simulate this $\hat{\mu}'$ by a subtree. We terminate the recursion after enough steps, and use a basic star gadget which is closest to the desired value as an approximation in the leaf. With a correlation decay argument, we show that the error in the root can be exponentially small in terms of the depth, although there may be a constant error in the leaf. A detailed construction with special treatment for the boundary cases are formally given in Algorithm 1. In the description of the algorithm and the analysis below, we denote $\alpha \triangleq \frac{\sqrt{\beta\gamma}-1}{\sqrt{\beta\gamma}+1} < 1$.

Before we prove that the construction is correct, we obtain a few observations which will be used in the proof. The condition on μ in Lemma 8 is due to the following property we need.

Algorithm 1: Constructing G_ℓ

function `construct`$(\ell, \hat{\mu})$:

input : Recursion depth ℓ; Target $0 < \hat{\mu} \le \mu^*$ to simulate;
output : Graph G_ℓ constructed.
begin
  if $\ell = 0$ **then**
    Let k be the positive integer such that $\mu h(\mu)^{k+1} < \hat{\mu} \le \mu h(\mu)^k$;
    return S_k;
  else
    Let k be the non-negative integer with $\mu^* h(\mu)^{k+1} < \hat{\mu} \le \mu^* h(\mu)^k$;
    $\mathcal{Y}' \leftarrow k \cdot \mathcal{S}_0$; `// a set of` k `copies of` $\mathcal{S}_0$`.`
    $\mu_1 \leftarrow \frac{\hat{\mu}}{h(\mu)^k}$;
    `// Invariant:` $\mu h(x')^{d-i+1} = \mu_i$ `has a solution` $0 < x' \le \mu^*$`.`
    for $i \leftarrow 1$ **to** $d-1$ **do**
      if $\mu h(\mu^*) h(0)^{d-i} \ge \mu_i$ **then**
        $y_i \leftarrow 0$; $w \leftarrow \lfloor \frac{\ell \cdot \ln \alpha - \ln(d\mu)}{\ln \beta} \rfloor + 1$; $Y_i \leftarrow \mathcal{S}_w$;
      else
        $y_i \leftarrow \mu^*$; $t \leftarrow \lfloor \frac{\ell \cdot \ln \alpha - \ln d - \ln \iota}{\ln c} \rfloor + 1$; $Y_i \leftarrow \mathcal{T}_t$;
      $\mu_{i+1} \leftarrow \frac{\mu_i}{h(y_i)}$;
    Let $\hat{\mu}'$ be the solution of $\mu h(x) = \mu_d$ in $(0, \mu^*]$;
    $\mathcal{Y} \leftarrow \mathcal{Y}' \cup \{Y_i\}_{i \ge 1}^{d-1}$;
    $\delta \leftarrow \exp(-\frac{\ln \gamma \ln \alpha}{\ln \beta} \ell + \frac{\ln \gamma \ln(d\mu)}{\ln \beta} + \ln \frac{\mu}{\gamma})$;
    if $\hat{\mu}' \le \delta$ **then**
      Choose the largest integer w such that $\mu \left(\frac{1}{\gamma}\right)^w > \delta$;
      return `comb`$(\mathcal{Y} \cup \{\mathcal{S}_w\})$;
    else
      return `comb`$(\mathcal{Y} \cup$ `construct`$(\ell - 1, \hat{\mu}'))$;

▶ **Proposition 14.** *Let* $\mu > \frac{\gamma^d}{\beta}(\beta\gamma - 1)\left(1 + \frac{d+1}{\ln(\beta(\beta\gamma)^d)}\right)$*, for any* μ_1 *with* $\mu^* h(\mu) < \mu_1 \le \mu^*$*, the equation* $\mu h(x)^d = \mu_1$ *always has a solution with* $0 < x \le \mu^*$*.*

Proof. It suffices to show $\mu \cdot h(0)^d \le \mu^* h(\mu)$ and $\mu \cdot h(\mu^*)^d \ge \mu^*$. Since $\mu^* = \mu h(\mu^*)^d$, the second part is trivial. As for the first part, it is sufficient to show $\left(\frac{h(\mu^*)}{h(0)}\right)^d h(\mu) > 1$. Note that $\left(\frac{h(\mu^*)}{h(0)}\right)^d h(\mu) > \gamma^d h(\mu^*)^{d+1} > \gamma^d \left(\beta - \frac{\beta\gamma - 1}{\mu^*}\right)^{d+1}$,

$$\gamma^d \left(\beta - \frac{\beta\gamma - 1}{\mu^*}\right)^{d+1} > 1 \iff \ln\left(\beta(\beta\gamma)^d\right) + (d+1)\ln\left(1 - \frac{\beta\gamma - 1}{\beta\mu^*}\right) > 0,$$

$$(d+1)\ln\left(1 - \frac{\beta\gamma - 1}{\beta\mu^*}\right) \overset{(\clubsuit)}{>} -(d+1)\frac{\frac{\beta\gamma-1}{\beta\mu^*}}{1 - \frac{\beta\gamma-1}{\beta\mu^*}} \overset{(\spadesuit)}{>} -\ln\left(\beta(\beta\gamma)^d\right),$$

where $(\clubsuit)$ is due to $\ln(1-x) > -\frac{x}{1-x}$ for $x \in (0,1)$, and $(\spadesuit)$ is by the fact that $\beta(\beta\gamma)^d > 1$ and the choice of μ such that $-\frac{\beta\gamma-1}{\beta\mu^*} > \frac{\ln\left(\beta(\beta\gamma)^d\right)+d+1}{\beta\ln(\beta(\beta\gamma)^d)}$. ◀

▶ **Proposition 15.** *For every $x, t \geq 0$, it holds that $h(x+t) \leq (1+t)h(x)$ and $h\left((1+t)x\right) \leq (1+t)h(x)$.*

Proof. Note that $x, t \geq 0$,

$$h(x+t) \leq (1+t)h(x) \iff \left(\frac{\beta(x+t)+1}{x+t+\gamma}\right) \leq (1+t)\left(\frac{\beta x+1}{x+\gamma}\right)$$
$$\iff t^2(1+\beta x) + t\left(1+\gamma\left(1+\beta(x-1)\right)+x+\beta x^2\right) \geq 0.$$

Since $\left(1+\gamma\left(1+\beta(x-1)\right)+x+\beta x^2\right) > 0$, the inequality always holds.

$$h\left((1+t)x\right) \leq (1+t)h(x) \iff \frac{x(1+t)\beta+1}{x(1+t)+\gamma} \leq (1+t)\frac{\beta x+1}{x+\gamma}$$
$$\iff t^2(x+\beta x^2) + t(\gamma+2x+\beta x^2) \geq 0$$

Again every term is non-negative, the last inequality is always true. ◀

We first verify that the algorithm is well defined, namely $\mu h(x) = \mu_d$ does have a solution $\hat{\mu}'$ in $(0, \mu^*]$. This can be done by verifying the loop invariant "$\mu h(x')^{d-i+1} = \mu_i$ has a solution $0 < x' \leq \mu^*$" inductively.

Initialization. For $i = 1$, by Proposition 14, for some $0 < \tilde{x} \leq \mu^*$ it holds that $\mu h(\tilde{x})^{d-i+1} = \mu_i$.

Maintenance. Assuming $\mu h(\tilde{x})^{d-i+1} = \mu_i$ has solutions $\tilde{x} \in (0, \mu^*]$, we verify that $\mu h(x')^{d-i} = \mu_{i+1} \equiv \frac{\mu_i}{h(y_i)}$ has solutions $x' \in (0, \mu^*]$ for $i \in [1, d-1]$.

Case $\mu h(\mu^*)h(0)^{d-i} \geq \mu_i$. By assumption we have $\mu h(0)^{d-i+1} < \mu_i$, also note that $\mu_i \leq \mu h(\mu^*)h(0)^{d-i} \leq \mu h(0)h(\mu^*)^{d-i}$, hence $\mu h(0)^{d-i} < \frac{\mu_i}{h(0)} \leq \mu h(\mu^*)^{d-i}$. Then by continuity, $\mu h(x')^{d-i} = \frac{\mu_i}{h(0)}$ has solutions $0 < x' \leq \mu^*$.

Case $\mu h(\mu^*)h(0)^{d-i} < \mu_i$. By assumption $\mu h(\mu^*)^{d-i+1} \geq \mu_i$, thus $\mu h(0)^{d-i} < \frac{\mu_i}{h(\mu^*)} \leq \mu h(\mu^*)^{d-i}$, hence $\mu h(x')^{d-i} = \frac{\mu_i}{h(\mu^*)}$ has solutions $0 < x' \leq \mu^*$.

Termination. After the loop completes, $\mu h(x') = \mu_d$ has solutions $0 < x' \leq \mu^*$.

Now we verify the vertex weight gadget returned by the construction satisfies our requirement by choosing $\ell = O(-\log \varepsilon)$.

▶ **Lemma 16.** *For $0 < \hat{\mu} \leq \mu^*(\beta, \gamma, \mu)$, and let $G(V, E)$ be the graph returned by* `construct`$(\ell, \hat{\mu})$*, we have the following:*

1. $\exp(-(c+\ell)\cdot\alpha^\ell) \leq \frac{\mu(G)}{\hat{\mu}} \leq \exp((c+\ell)\cdot\alpha^\ell)$*, where $c = \ln\gamma$ and $\alpha = \frac{\sqrt{\beta\gamma}-1}{\sqrt{\beta\gamma}+1} < 1$;*
2. $|G| = \exp(O(\ell))$.

Proof. We apply induction on ℓ for both statements. We prove for (1) first. For the base case $\ell = 0$, we have

$$|\ln\mu(G) - \ln\hat{\mu}| \leq \left|\ln\mu h(\mu)^k - \ln\mu h(\mu)^{k+1}\right| = -\ln h(\mu) \leq \ln\gamma.$$

Assume that the statement holds for smaller ℓ. Let k, $\{y_i\}_{1\leq i\leq d-1}$ and $\{Y_i\}_{1\leq i\leq d-1}$ be parameters chosen in the algorithm. Define

$$F(z) = \ln\left(\mu h(\mu)^k \prod_{i=1}^{d-1} h(y_i)h(\exp(z))\right), \tilde{F}(z) = \ln\left(\mu h(\mu)^k \prod_{i=1}^{d-1} h(\mu(Y_i))h(\exp(z))\right).$$

We note that $F(z)$ is the *correct* recursion to compute $\ln(\mu(G))$ and $\tilde{F}(z)$ is our *approximate* recursion used in the algorithm.

In the following, we distinguish between $\hat{\mu}' \leq \delta$ and $\hat{\mu}' > \delta$.

- If $\hat{\mu}' \le \delta$, then $\ln \mu(G) = \tilde{F}(\ln \mu(\mathcal{S}_w))$ and $\ln \hat{\mu} = F(\ln \hat{\mu}')$. We have

$$\begin{aligned} F(\ln \hat{\mu}') \le \tilde{F}(\ln \mu(\mathcal{S}_w)) &= \ln \left(\mu h(\mu)^k \prod_{i=1}^{d-1} h(\mu(Y_i)) h(\mu(\mathcal{S}_w)) \right) \\ &\overset{(\heartsuit)}{\le} \alpha^\ell + \ln \left(\mu h(\mu)^k \prod_{i=1}^{d-1} h(y_i) h(\hat{\mu}') \right) \\ &= \alpha^\ell + F(\ln \hat{\mu}')), \end{aligned}$$

  where $(\heartsuit)$ follows from the following facts derived from Proposition 15:

  (i) If $y_i = 0$, then $0 \le \mu(Y_i) \le \frac{\alpha^\ell}{d}$, which implies $h(\mu(Y_i)) \ge h(y_i)$ and $h(\mu(Y_i)) \le \left(1 + \frac{\alpha^\ell}{d}\right) h(y_i) \le \exp\left(\frac{\alpha^\ell}{d}\right) h(y_i)$.

  (ii) If $y_i = \mu^*$, then $\mu^* \le \mu(Y_i) \le \exp\left(\frac{\alpha^\ell}{d}\right) \mu^*$, which implies $h(\mu(Y_i)) \ge h(y_i)$ and $h(\mu(Y_i)) \le \exp\left(\frac{\alpha^\ell}{d}\right) h(y_i)$.

  (iii) We claim that $\hat{\mu}' < \mu \left(\frac{1}{\gamma}\right)^w \le \mu(\mathcal{S}_w) \le \mu\beta^w \le \frac{\alpha^\ell}{d}$. The only nontrivial part is to verify that $\mu\beta^w \le \frac{\alpha^\ell}{d}$. Since w is the largest integer that $\hat{\mu}' < \mu \left(\frac{1}{\gamma}\right)^w$, we have $\mu \left(\frac{1}{\gamma}\right)^{w+1} \le \hat{\mu}'$, which gives $w \ge \frac{\ln \mu - \ln \delta}{\ln \gamma} - 1$. Plug this into $\mu\beta^w \le \frac{\alpha^\ell}{d}$ and let $\delta = \exp(-\frac{\ln \gamma \ln \alpha}{\ln \beta} \ell + \frac{\ln \gamma \ln(d\mu)}{\ln \beta} + \ln \frac{\mu}{\gamma})$, the inequality holds. Thus $h(\mu(\mathcal{S}_w)) \ge h(\hat{\mu}')$ and $h(\mu(\mathcal{S}_w)) \le h(\frac{\alpha^\ell}{d}) \le \left(1 + \frac{\alpha^\ell}{d}\right) h(\hat{\mu}') \le \exp\left(\frac{\alpha^\ell}{d}\right) h(\hat{\mu}')$.

- If $\hat{\mu}' > \delta$, define $x = \mu(\texttt{construct}(\ell - 1, \hat{\mu}'))$, then by induction hypothesis, it holds that $|\ln x - \ln \hat{\mu}'| \le (c + (\ell - 1)) \cdot \alpha^{\ell - 1}$.
  Then similarly by Proposition 15 and the choice of w and t, we have $F(\ln x) \le \tilde{F}(\ln x) \le F(\ln x) + \alpha^\ell$. Thus by construction, we have

$$\begin{aligned} |\ln \mu(G) - \ln \hat{\mu}| &= \left|\tilde{F}(\ln x) - F(\ln \hat{\mu}')\right| \\ &\le \alpha^\ell + |F(\ln x) - F(\ln \hat{\mu}')| \\ &\le \alpha^\ell + |F'(\ln \tilde{x})| \cdot |\ln x - \ln \hat{\mu}'| \quad \text{(for some } \tilde{x} \in [\hat{\mu}', x].) \\ &\le \alpha^\ell + (\ell - 1) |F'(\ln \tilde{x})| \, \alpha^{\ell - 1} + c |F'(\ln \tilde{x})| \, \alpha^{\ell - 1} \end{aligned}$$

  Thus it is sufficient to show that $|F'(\ln \tilde{x})| \le \alpha$. In fact, $F'(\ln x) = \frac{x \cdot h'(x)}{h(x)} = \frac{(\beta\gamma - 1)x}{(x + \gamma)(\beta x + 1)} \le \frac{\beta\gamma - 1}{\left(\sqrt{\beta\gamma} + 1\right)^2} = \alpha$.

Now we prove (2) of the Lemma. We denote $s(\ell) = \max_{\hat{\mu}} |\texttt{construct}(\ell, \hat{\mu})|$ and show that $s(\ell) = \ell \exp(O(\ell)) = \exp(O(\ell))$.

If $\ell = 0$, since $\hat{\mu}$ is either the eventual external field (which is a constant bounded away from 0), or $\hat{\mu} > \delta$, we have $s(\ell) = |\mathcal{S}_k| = O(1)$.

If $\ell > 0$, then $|Y_i| = \exp(O(\ell))$ and thus $|\mathcal{Y}| = \exp(O(\ell))$. By our choice of δ, it holds that $w = O(\ell)$ and thus $|S_w| = O(\ell)$. Therefore,

$$s(\ell) = \exp(O(\ell)) + \max\{s(\ell - 1), O(\ell)\} = \ell \exp(O(\ell)) = \exp(O(\ell)).$$

This concludes the proof. ◀

3.2 The $\beta > 1$ case

The $\beta > 1$ case follows a similar argument as that in [7] and is known as a folklore. We include a formal proof here to be self-contained.

▶ **Theorem 17.** *Let $\gamma > \beta > 1$ and $\mu > \frac{\gamma-1}{\beta-1}$. Then* $\text{Ferro}(\beta, \gamma, \mu)$ *is* #BIS*-hard.*

We follow the same idea of simulating external field and make use of Theorem 3. In the case $\beta > 1$, it is easy to see that we can simulate all positive external fields.

▶ **Lemma 18.** *For every $\hat{\mu} > 0$, there is a family of vertex weight gadgets $\{G_m\}_{m\geq 1}$ that realizes $\hat{\mu}$. Moreover, G_m is constructible in time $m^{O(1)}$ and*

$$\exp(-\frac{1}{m}) \leq \frac{\mu(G_m)}{\hat{\mu}} \leq \exp(\frac{1}{m}). \tag{1}$$

Proof. For any $m \geq 1$, we add x self-loops and y bristles to a single vertex v, where x and y are integers to be determined. Let v be the output of G_m, then $\mu(G_m) = \mu \left(\frac{\beta}{\gamma}\right)^x \left(\frac{\mu\beta+1}{\mu+\gamma}\right)^y$. Denote $a = \ln \frac{\gamma}{\beta}$, $b = \ln \frac{\mu\beta+1}{\mu+\gamma}$ and $c = \frac{\ln \hat{\mu}}{\ln \mu}$, then (1) is equivalent to

$$|(y \cdot b - x \cdot a) - c| \leq \frac{1}{m}.$$

We can use a procedure similar to extended Euclidean algoroithm to find such integers x, y in time $O(\ln m)$, such that it also guarantees $x, y = m^{O(1)}$. ◀

4 Improved Tractable Result

In this section, we establish the following tractable result:

▶ **Theorem 19.** *Let $\beta < \gamma$, $\beta\gamma > 1$ and $\mu \leq \gamma/\beta$. Then there is an FPRAS for* $\text{Ferro}(\beta, \gamma, \mu)$.

The proof of this theorem follows by refining the proof in [9], where they establish the tractable result for $\mu \leq (\gamma/\beta)^{\delta/2}$ for δ being the *minimum* degree of vertices in the graph. Specifically, we first contract all vertices with degree one and modify the external fields of their neighboring vertices, this only scales the partition function by a constant. Next, just as in [9], we shall reduce a (β, γ, μ) instance to a ferromagnetic Ising instance and apply the following celebrated result, which is first introduced in [10] for uniform external fields and refined for non-uniform external fields in [9]:

▶ **Theorem 20** ([10] and [9])**.** *There is an FPRAS for Ising system $(a, a, \mathcal{V})$ provided that $a > 1$ and all external fields in $\mathcal{V}$ are at most one.*

Let $G(V, E)$ be an instance of (β, γ, μ) system, we repeatedly apply the following operations until no degree one vertices can be found:

1. Pick a vertex u of degree one. Denote its incident edge by $e = (u, v)$. Let μ_u and μ_v be external fields on u and v respectively.
2. Remove u and edge (u, v), update $\mu_v \leftarrow \mu_v h(\mu_u)$.

Let $G'(V', E')$ be the remaining graph. G' either has no vertices of degree one, or it only contains a single vertex. Moreover, for every $v \in V'$, the external fields μ'_v satisfies $\mu'_v \leq \mu$. This can be easily verified given that $\mu \leq \gamma/\beta$. Let $\mathcal{U} = \{\mu'_v \mid v \in V'\}$, consider G' as an instance of $(\beta, \gamma, \mathcal{U})$ system, clearly $Z_{(\beta,\gamma,\mu)}(G) = Z^* \cdot Z_{(\beta,\gamma,\mathcal{U})}(G')$ where Z^* is an easily polynomial-time computable factor.

Let $\mathcal{V} = \left\{ \mu'_v \left(\frac{\beta}{\gamma}\right)^{d_v/2} \middle| v \in V' \right\}$ where d_v is the degree of v in G'. Let $\hat{G}(\hat{V}, \hat{E})$ be a copy of G' with $\hat{\mu}_v = \mu'_v \left(\frac{\beta}{\gamma}\right)^{d_v/2}$ for every $v \in \hat{V}$. We are going to verify that $Z_{(\beta,\gamma,\mathcal{U})}(G') = \sqrt{\frac{\gamma}{\beta}}^{|E'|} \cdot Z_{(a,a,\mathcal{V})}(\hat{G})$ for $a = \sqrt{\beta\gamma}$.

Define $A = \begin{bmatrix} \beta & 1 \\ 1 & \gamma \end{bmatrix}$, $A' = \begin{bmatrix} \gamma & \sqrt{\gamma/\beta} \\ \sqrt{\gamma/\beta} & \gamma \end{bmatrix}$ and $\hat{A} = \begin{bmatrix} \sqrt{\beta\gamma} & 1 \\ 1 & \sqrt{\beta\gamma} \end{bmatrix}$. Then,

$$
\begin{aligned}
Z_{(\beta,\gamma,\mathcal{U})}(G') &= \sum_{\sigma \in \{0,1\}^{V'}} \prod_{(u,v)\in E'} A_{\sigma_u,\sigma_v} \prod_{v \in V'} {\mu'}_v^{1-\sigma_v} \\
&= \sum_{\sigma \in \{0,1\}^{V'}} \prod_{(u,v)\in E'} A'_{\sigma_u,\sigma_v} \prod_{v \in V'} \left(\left(\sqrt{\frac{\beta}{\gamma}} \right)^{d_v} {\mu'}_v \right)^{1-\sigma_v} \\
&= \sqrt{\frac{\gamma}{\beta}}^{|E'|} \sum_{\sigma \in \{0,1\}^{V'}} \prod_{(u,v)\in E'} \hat{A}_{\sigma_u,\sigma_v} \prod_{v \in V'} \left(\left(\sqrt{\frac{\beta}{\gamma}} \right)^{d_v} {\mu'}_v \right)^{1-\sigma_v} \\
&= \sqrt{\frac{\gamma}{\beta}}^{|E'|} \sum_{\sigma \in \{0,1\}^{\hat{V}}} \prod_{(u,v)\in \hat{E}} \hat{A}_{\sigma_u,\sigma_v} \prod_{v \in V'} \hat{\mu}_v^{1-\sigma_v} \\
&= \sqrt{\frac{\gamma}{\beta}}^{|E'|} \cdot Z_{(a,a,\mathcal{V})}(\hat{G})
\end{aligned}
$$

Finally, to apply Theorem 20, we only need $\hat{\mu}_v \leq 1$ for all $v \in \hat{V}$. Recall that $\hat{G}$ has $\delta \geq 2$, hence $\mu \leq \gamma/\beta$ implies $\hat{\mu}_v \leq 1$. This concludes the proof.

References

1. Antar Bandyopadhyay and David Gamarnik. Counting without sampling: Asymptotics of the log-partition function for certain statistical physics models. *Random Structures & Algorithms*, 33(4):452–479, 2008.
2. Andrei Bulatov and Martin Grohe. The complexity of partition functions. *Theoretical Computer Science*, 348(2):148–186, 2005.
3. Jin-Yi Cai, Andreas Galanis, Leslie Ann Goldberg, Heng Guo, Mark Jerrum, Daniel Štefankovic, and Eric Vigoda. #BIS-hardness for 2-spin systems on bipartite bounded degree graphs in the tree nonuniqueness region. *To Appear in RANDOM 2014*, 2014.
4. Martin Dyer, Leslie Ann Goldberg, Catherine Greenhill, and Mark Jerrum. The relative complexity of approximate counting problems. *Algorithmica*, 38(3):471–500, 2004.
5. Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. An approximation trichotomy for boolean #CSP. *Journal of Computer and System Sciences*, 76(3):267–277, 2010.
6. Andreas Galanis, Daniel Štefankovic, and Eric Vigoda. Inapproximability of the partition function for the antiferromagnetic Ising and hard-core models. *arXiv preprint arXiv:1203.2226*, 2012.
7. Leslie Ann Goldberg and Mark Jerrum. The complexity of ferromagnetic Ising with local fields. *Combinatorics, Probability and Computing*, 16(01):43–61, 2007.
8. Leslie Ann Goldberg and Mark Jerrum. Approximating the partition function of the ferromagnetic Potts model. *Journal of the ACM (JACM)*, 59(5):25, 2012.
9. Leslie Ann Goldberg, Mark Jerrum, and Mike Paterson. The computational complexity of two-state spin systems. *Random Structures & Algorithms*, 23(2):133–154, 2003.

10 Mark Jerrum and Alistair Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal on computing*, 22(5):1087–1116, 1993.

11 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.

12 Liang Li, Pinyan Lu, and Yitong Yin. Approximate counting via correlation decay in spin systems. In *Proceedings of the 23th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*, pages 922–940. SIAM, 2012.

13 Liang Li, Pinyan Lu, and Yitong Yin. Correlation decay up to uniqueness in spin systems. In *Proceedings of the 24th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'13)*, pages 67–84, 2013.

14 Chengyu Lin, Jingcheng Liu, and Pinyan Lu. A simple FPTAS for counting edge covers. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'14)*, pages 341–348, 2014.

15 Pinyan Lu, Menghui Wang, and Chihao Zhang. FPTAS for weighted Fibonacci gates and its applications. In *Proceedings of the 41th International Colloquium on Automata, Languages and Programming (ICALP'14, Track A)*, pages 787–799, 2014.

16 Alistair Sinclair, Piyush Srivastava, and Marc Thurley. Approximation algorithms for two-state anti-ferromagnetic spin systems on bounded degree graphs. In *Proceedings of the 23th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*, pages 941–953. SIAM, 2012.

17 Allan Sly. Computational transition at the uniqueness threshold. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS'10)*, pages 287–296. IEEE, 2010.

18 Allan Sly and Nike 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'12)*, pages 361–369. IEEE, 2012.

19 Eric Vigoda. Improved bounds for sampling colorings. *Journal of Mathematical Physics*, 41(3):1555–1569, 2000.

20 Dror Weitz. Counting independent sets up to the tree threshold. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC'06)*, pages 140–149. ACM, 2006.

It's a Small World for Random Surfers*†

Abbas Mehrabian[1] and Nick Wormald[2]

1 **Department of Combinatorics and Optimization, University of Waterloo**
Waterloo, ON, Canada
amehrabi@uwaterloo.ca

2 **School of Mathematical Sciences, Monash University**
Clayton, VIC, Australia
nick.wormald@monash.edu

Abstract

We prove logarithmic upper bounds for the diameters of the random-surfer Webgraph model and the PageRank-based selection Webgraph model, confirming the small-world phenomenon holds for them. In the special case when the generated graph is a tree, we get close lower and upper bounds for the diameters of both models.

1998 ACM Subject Classification C.2.1 Network Architecture and Design, G.2.2 Graph Theory, G.3 Probability and Statistics

Keywords and phrases random-surfer webgraph model, PageRank-based selection model, small-world phenomenon, height of random trees, probabilistic analysis, large deviations

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.857

1 Introduction

Due to the ever growing interest in social networks, the Webgraph, biological networks, etc., in recent years a great deal of research has been built around modelling real world networks (see, e. g., the monographs [6, 8, 10, 15]). One of the important observations about many real world networks involves the diameter, which is the maximum shortest-path distance between any two nodes. The so-called *small-world phenomenon* is that the diameter of a network is significantly smaller than its size, typically growing as a polylogarithmic function.

The Webgraph is a directed graph whose vertices are the static web pages, and there is an edge joining two vertices if there is a hyperlink in the first page pointing to the second page. Barabási and Albert [1] in 1999 introduced one of the first models for the Webgraph, widely known as the *preferential attachment model*. Their model can be informally described as follows (see [5] for the formal definition). Let d be a positive integer. We start with a fixed small graph, and in each time-step a new vertex appears and is joined to d old vertices, where the probability of joining to each old vertex is proportional to its *degree*. Pandurangan, Raghavan and Upfal [20] in 2002 introduced the *PageRank-based selection model* for the Webgraph. This model is similar to the previous model, except the attachment probabilities are proportional to the *PageRanks* of the vertices rather than their degrees (see Section 2 for the formal definition). Blum, Chan, and Rwebangira [4] in 2006 introduced a *random-surfer model* for the Webgraph, in which the d out-neighbours of the new vertex are chosen by doing d independent random walks that start from random vertices and whose lengths are

* The first author was supported by the Vanier Canada Graduate Scholarships program, and the second author was supported by Australian Laureate Fellowships grant FL120100125.

† The full version of this paper is available in http://arxiv.org/abs/1404.7189.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 857–871

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

geometric random variables with parameter p (see Section 2 for the formal definition). It was shown that under certain conditions (see Section 2), the previous two models are equivalent.

The directed models considered here generate directed acyclic graphs (new vertices create edges to old vertices), so it is natural to define the *diameter* of a directed graph as the maximum shortest-path distance between any two vertices in its underlying undirected graph. The diameter of the preferential attachment model was analysed by Bollobás and Riordan [5]. Previous work on the PageRank-based selection and random-surfer models has focused on their degree distributions. To the best of our knowledge, the diameters of these models have not been studied previously, and it is an open question even whether they have logarithmic diameter. One of the main contributions of this paper is giving logarithmic upper bounds for their diameters. We also give close lower and upper bounds in the special case $d = 1$, namely when the generated graph is (almost) a tree. It turns out that the key parameter in this case is the *height* of the generated random tree. We find the asymptotic value of the height for all $p \in [0.21, 1]$, and for $p \in (0, 0.21)$ we provide logarithmic lower and upper bounds. Our results hold *asymptotically almost surely (a.a.s.)*, which means the probability that they are true approaches 1 as the number of vertices grows. As the two models are equivalent, we will focus on the random-surfer model, since it is easier to work with.

1.1 Our Approach and Organization of the Paper

In the preferential attachment model and most of its variations (see, e. g., [1, 13, 14, 17]) the probability that the new vertex attaches to an old vertex v, called the *attraction* of v, is proportional to a deterministic function of the degree of v. In other variations (see, e. g., [3, 16]) the attraction also depends on the so-called 'fitness' of v, which is a random variable generated independently for each vertex and does not depend on the structure of the graph. For analysing such models when they generate trees, a typical technique is to approximate them with population-dependent branching processes and prove that results on the corresponding branching processes carry over to the original models. A classical example is Pittel [22] who estimated the height of random recursive trees. Bhamidi [2] used this technique to show that the height of a variety of preferential attachment trees is asymptotic to a constant times the logarithm of the number of vertices, where the constant depends on the parameters of the model.

In the random-surfer Webgraph model, however, the attraction of a vertex does not depend only on its degree, but rather on the graph's general structure, so the branching processes techniques cannot apply directly, and new ideas are needed.

The crucial novel idea in our proof is to reduce the attachment rule to a simple one, with the help of introducing (possibly negative) 'weights' for the edges. First, consider the general case, $d \geq 1$. Whenever a new vertex appears, it builds d new edges to old vertices; suppose that we mark the first new edge. Then the marked edges induce a spanning tree whose diameter we bound, and thus we get an upper bound for the diameter of the random-surfer Webgraph model.

In the special case $d = 1$, we obtain a *random recursive tree* with edge weights, and then we adapt a powerful technique developed by Broutin and Devroye [7] (that uses branching processes) to study its weighted height. This technique is based on large deviations. The main theorem of Broutin and Devroye [7, Theorem 1] is not applicable here for two reasons. Firstly, the weights of edges on the path from the root to each vertex are not independent, and secondly, the weights can be negative.

We define the models and state our main results in Section 2. In Section 3 we prove Theorem 2, giving a logarithmic upper bound for the diameter of the random-surfer Webgraph

model in the general case $d \geq 1$. In Sections 4–6 we focus on the special case $d = 1$ and we prove Theorems 3 and 4. Section 4 contains the main technical contribution of this paper, where we explain how to transform the random-surfer tree model into one that is easier to analyse. The lower and upper bounds are proved in Sections 5 and 6, respectively. Concluding remarks appear in Section 7. All proofs omitted from this extended abstract can be found in the full version.

2 Definitions and Main Results

Given $p \in (0, 1]$, let $\mathrm{Geo}(p)$ denote a geometric random variable with parameter p; namely for every nonnegative integer k, $\mathbb{P}[\mathrm{Geo}(p) = k] = (1-p)^k p$.

▶ **Definition 1** (Random-Surfer Webgraph Model [4])**.** Let d be a positive integer and let $p \in (0, 1]$. Generate a random directed rooted n-vertex multigraph, with all vertices having out-degree d. Start with a single vertex v_0, the root, with d self-loops. At each subsequent step s, where $1 \leq s \leq n-1$, a new vertex v_s appears and d edges are created from it to vertices in $\{v_0, v_1, \ldots, v_{s-1}\}$, by doing the following probabilistic procedure d times, independently: choose a vertex u uniformly at random from $\{v_0, v_1, \ldots, v_{s-1}\}$, and a fresh random variable $X = \mathrm{Geo}(p)$; perform a simple random walk of length X starting from u, and join v_s to the last vertex of the walk. Note that the random walk is performed on the *directed* graph.

The motivation behind this definition is as follows. Think of the vertex v_s as a new web page that is being set up. Say the owner wants to put d links in her web page. To build each link, she does the following: she goes to a random page. With probability p she likes the page and puts a link to that page. Otherwise, she clicks on a random link on that page, and follows the link to a new page. Again, with probability p she likes the new page and puts a link to that, otherwise clicks on a random link etc., until she finds a desirable page to link to. The geometric random variables correspond to this selection process.

Our main result regarding the diameter of the random-surfer Webgraph model is the following theorem (recall that the diameter of a directed graph is defined as the diameter of its underlying undirected graph).

▶ **Theorem 2.** *Let d be a positive integer and let $p \in (0, 1]$. A.a.s. as $n \to \infty$ the diameter of the random-surfer Webgraph model with parameters p and d is at most $8e^p(\log n)/p$.*

Notice that the upper bound in Theorem 2 does not depend on d (whereas one would expect that the diameter must decrease asymptotically as d increases). This independence is because in our argument we employ only the first edge created by each new vertex to bound the diameter. Obtaining better bounds as d grows is related to analysing the first order statistic of several intricate random variables, and seems to be much harder.

▶ Remark. In considering the undirected version of diameter of directed graphs, we follow [18]. The directed version of diameter, i. e. the largest directed distance between any two vertices, is much harder to study for the following reason. Let v_s be a new vertex, and let u be the start vertex of a corresponding random walk. The random walk from u could potentially move to the *worst* neighbour of u, i. e. a neighbour whose directed depth is much larger than u, causing a great increase in the directed depth of v_s (compared to the case that it attaches to u). However, in the case of undirected distances, each step of the walk can increase any undirected distance by at most 1.

A *random-surfer tree* is an undirected tree obtained from a random-surfer Webgraph with $d = 1$ by deleting the self-loops of the root and ignoring the edge directions. The *height*

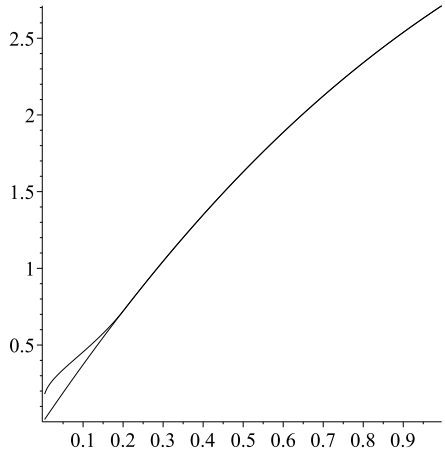


Figure 1 The functions c_L and c_U in Theorems 3 and 4.

of a tree is defined as the maximum graph distance between a vertex and the root. Our main result regarding the height of the random-surfer tree model is the following theorem.

▶ **Theorem 3.** *For $p \in (0,1)$, let s be the unique solution in $(0,1)$ to*

$$s \log\left(\frac{(1-p)(2-s)}{1-s}\right) = 1\,. \tag{1}$$

Let $p_0 \approx 0.206$ be the unique solution in $(0,1/2)$ to

$$\log\left(\frac{1-p}{p}\right) = \frac{1-p}{1-2p}\,. \tag{2}$$

Define the functions $c_L, c_U : (0,1) \to \mathbb{R}$ as

$$c_L(p) = \exp(1/s)s(2-s)p\,,$$

and

$$c_U(p) = \begin{cases} c_L(p) & \text{if } p_0 \le p < 1 \\ \left(\log\left(\frac{1-p}{p}\right)\right)^{-1} & \text{if } 0 < p < p_0\,. \end{cases}$$

For every fixed $\varepsilon > 0$, a.a.s. as $n \to \infty$ the height of the random-surfer tree model with parameter p is between $(c_L(p) - \varepsilon)\log n$ and $(c_U(p) + \varepsilon)\log n$.

It is easy to see that the value p_0 and the functions c_L and c_U (plotted in Figure 1) are well defined. Also, c_L and c_U are continuous, and $\lim_{p\to 0} c_L(p) = \lim_{p\to 0} c_U(p) = 0$ and $\lim_{p\to 1} c_L(p) = e$. We suspect that the gap between our bounds when $p < p_0$ is an artefact of our proof technique, and we do not expect a phase transition in the behaviour of the height at $p = p_0$.

We also prove lower and upper bounds for the diameter, which are close to being tight.

▶ **Theorem 4.** *Let c_L and c_U be defined as in Theorem 3. For every fixed $\varepsilon > 0$, a.a.s. as $n \to \infty$ the diameter of the random-surfer tree model with parameter $p \in (0,1)$ is between $(2c_L(p) - \varepsilon)\log n$ and $(2c_U(p) + \varepsilon)\log n$.*

Immediately, we have the following corollary.

▶ **Corollary 5.** *Let c_L and p_0 be defined as in Theorem 3. For any $p \in [p_0, 1)$, the height of the random-surfer tree model with parameter p is a.a.s. asymptotic to $c_L(p) \log n$ as $n \to \infty$, and its diameter is a.a.s. asymptotic to $2c_L(p) \log n$.*

We remark that Theorem 4 does not imply Theorem 2; in fact it does not even imply the diameter of the random-surfer Webgraph model (with $d > 1$) is logarithmic. The reason is that in the random-surfer tree model, due to the tree structure, the random walk for each vertex always moves closer to the root, whereas in the random-surfer Webgraph model, this is not the case, and the random walk could move further from the root.

We now define the PageRank-based selection model introduced in [20, 21].

▶ **Definition 6** (PageRank and the PageRank-based Selection Webgraph Model [20, 21])**.** Let d be a positive integer and let $p, \beta \in [0, 1]$. The *PageRank* of a directed graph is a probability distribution over its vertices, which is the stationary distribution of the following random walk. The random walk starts from a vertex chosen uniformly at random. In each step, with probability p it jumps to a vertex chosen uniformly at random, and with probability $1 - p$ it walks to a random out-neighbour of the current vertex.

We generate a random n-vertex directed multigraph with all vertices having out-degree d. We start with a single vertex with d self-loops. At each subsequent step a new vertex appears, chooses d old vertices and attaches to them (where a vertex can be chosen multiple times). These choices are independent and the head of each edge is a uniformly random vertex of the existing graph with probability β, and is a vertex chosen according to the PageRank distribution on the existing graph with probability $1 - \beta$.

The motivation behind this definition is as follows. Consider the case $\beta = 0$. Think of the vertex v_s as a new web page that is being set up. Say the owner wants to put d links in her web page. She finds the destination pages using d independent Google searches. Since Google sorts the search results according to their PageRank (see [19]), under some suitable randomness assumptions, we may imagine that the probability that a given page is linked to is close to its PageRank.

Chebolu and Melsted [9] showed the PageRank-based selection Webgraph model with $\beta = 0$ is equivalent to the random-surfer Webgraph model. Hence the conclusions of Theorems 2–4 apply to the former model with $\beta = 0$. Moreover, the proof of Theorem 2 easily extends to the PageRank-based selection Webgraph model, giving the same conclusion for all $\beta \in [0, 1]$

In Theorems 3 and 4 we have assumed that $p < 1$, since the situation for $p = 1$ has been clarified in previous work. Let $p = 1$. Then a random-surfer tree has the same distribution as a so-called random recursive tree, the height of which is a.a.s. asymptotic to $e \log n$ as proved by Pittel [22]. It is not hard to alter the argument in [22] to prove that the diameter is a.a.s. asymptotic to $2e \log n$. The diameter of a random-surfer Webgraph thus has also an asymptotically almost sure upper bound of $2e \log n$. For the rest of the paper, we fix $p \in (0, 1)$.

We will need two large deviation inequalities. Define the function $\Upsilon : (0, \infty) \to \mathbb{R}$ as

$$\Upsilon(x) = \begin{cases} x - 1 - \log(x) & \text{if } 0 < x \leq 1 \\ 0 & \text{if } 1 < x\,. \end{cases} \tag{3}$$

The following proposition follows easily from Cramér's Theorem (see, e. g., [11, Theorem 2.2.3, p. 27]) and the calculations in [7, p. 279].

▶ **Proposition 7.** *Let $E_1, E_2, \ldots, E_m$ be independent exponential random variables with mean 1. For any fixed $x > 0$, as $m \to \infty$ we have*

$$\exp\left(-\Upsilon(x)m - o(m)\right) \leq \mathbb{P}\left[E_1 + E_2 + \cdots + E_m \leq xm\right] \leq \exp(-\Upsilon(x)m)\,.$$

We include some definitions here. We define the *depth* of a vertex as the length of a shortest path (ignoring edge directions) connecting the vertex to the root, and the *height* of a graph G, denoted by $\mathrm{ht}(G)$, as the maximum depth of its vertices. Clearly the diameter is at most twice the height. In a weighted tree (a tree whose *edges* are weighted), we define the *weight* of a vertex to be the sum of the weights of the edges connecting the vertex to the root, and the *weighted height* of tree T, written $\mathrm{wht}(T)$, to be the maximum weight of its vertices. We view an unweighted tree as a weighted tree with unit edge weights, in which case the weight of a vertex is its depth, and the notion of weighted height is the same as the usual height.

3 Upper bound for the random-surfer Webgraph model

In this section we prove Theorem 2, giving an upper bound for the diameter of the random-surfer Webgraph model. Define the function $f : (-\infty, 1] \to \mathbb{R}$ as

$$f(x) = (2-x)^{2-x} p (1-p)^{1-x} (1-x)^{x-1}\,. \tag{4}$$

We will need a technical lemma which follows from straightforward calculations.

▶ **Lemma 8.** *Let η, c be positive numbers satisfying $\eta \geq 4e^p/p$ and $c \leq p\eta$. Then we have $-c\Upsilon(1/c) + c \log f(2 - \eta/c) < \eta(1-p)\log(1-p^3) - 1$.*

Proof of Theorem 2. We define an auxiliary weighted tree whose vertex set equals the vertex set of the graph generated by the random-surfer Webgraph model, and whose weighted height dominates the height of this graph. Then we bound the weighted height of this tree. Initially the tree has just one vertex v_0. Recall the growth of the random-surfer Webgraph model at each subsequent step $s \in \{1, 2, \ldots, n-1\}$: "a new vertex v_s appears and d edges are created from it to vertices in $\{v_0, v_1, \ldots, v_{s-1}\}$, by doing the following probabilistic procedure d times, independently: choose a vertex u uniformly at random from $\{v_0, v_1, \ldots, v_{s-1}\}$, and a fresh random variable $X = \mathrm{Geo}(p)$; perform a simple random walk of length X starting from u, and join v_s to the last vertex of the walk." Consider a step s and the first chosen $u \in \{v_0, \ldots, v_{s-1}\}$ and $X = \mathrm{Geo}(p)$. In the tree, we join the vertex v_s to u and set the weight of the edge $v_s u$ to be $X + 1$. Note that the edge weights are mutually independent. Clearly, the weight of v_s in the auxiliary tree is greater than or equal to the depth of v_s in the graph. Let $\eta = 4e^p/p$. Hence, to prove the theorem it suffices to show that a.a.s. the weighted height of the auxiliary tree is at most $\eta \log n$. We work with this tree in the rest of the proof.

We consider an alternative way to grow the tree, used in [12], which results in the same distribution. Let $U_1, U_2, \ldots$ be i.i.d. uniform random variables in $(0, 1)$. Then for each new vertex v_s, we join it to the vertex $v_{\lfloor sU_s \rfloor}$, which is indeed a vertex uniformly chosen from $\{v_0, \ldots, v_{s-1}\}$.

For convenience, we consider the tree when it has $n+1$ vertices $v_0, v_1, \ldots, v_n$. Define

$$\mathcal{A}(\ell) = n\mathbb{P}\left[D(n) = \ell\right] \mathbb{P}\left[W(n) > \eta \log n | D(n) = \ell\right]\,,$$

where $D(s)$ and $W(s)$ denote the depth and the weight of vertex v_s, respectively. We have

$$\mathbb{P}\left[\mathrm{wht}(\mathrm{tree}) > \eta \log n\right] \le \sum_{s=1}^{n} \mathbb{P}\left[W(s) > \eta \log n\right] \le n\mathbb{P}\left[W(n) > \eta \log n\right] = \sum_{\ell=1}^{n} \mathcal{A}(\ell)\,,$$

so to complete the proof it suffices to show $\sum_{\ell=1}^{n} \mathcal{A}(\ell) = o(1)$.

Let $P(0) = 0$ and for $s = 1, \dots, n$, let $P(s)$ denote the index of the parent of v_s. We have

$$\mathbb{P}\left[D(n) \ge \ell\right] = \mathbb{P}\left[D(P(n)) \ge \ell - 1\right] = \cdots = \mathbb{P}\left[D(P^{\ell-1}(n)) \ge 1\right] = \mathbb{P}\left[P^{\ell-1}(n) \ge 1\right]\,.$$

Since $P(m) = \lfloor mU_m \rfloor \le mU_m$ for each m and since the U_i are i.i.d., we have

$$\mathbb{P}\left[D(n) \ge \ell\right] = \mathbb{P}\left[P^{\ell-1}(n) \ge 1\right] \le \mathbb{P}\left[nU_1U_2\dots U_{\ell-1} \ge 1\right] \le \exp\left(-(\ell-1)\Upsilon\left(\frac{\log n}{\ell-1}\right)\right)\,, \tag{5}$$

where we have used Proposition 7 and the fact that $\log(1/U_i)$ is an exponential random variable with mean 1 for each i. The right-hand side is $o(1/n)$ for $\ell = 1.1e\log n$. Hence to complete the proof we need only show that

$$\mathcal{A}(\ell) = o(1/\log n) \qquad \forall \ell \in (0, 1.1e\log n)\,. \tag{6}$$

Fix an arbitrary positive integer $\ell \in (0, 1.1e\log n)$. The random variable $W(n)$, conditional on $D(n) = \ell$, is a sum of ℓ i.i.d. $1 + \mathrm{Geo}(p)$ random variables. By using Chernoff's technique of bounding the moment generating function of geometric random variables, we get

$$\mathbb{P}\left[W(n) > \eta \log n | D(n) = \ell\right] \le f(2 - \eta\log n/\ell)^{\ell}\,, \tag{7}$$

where f is defined in (4). Combining (5) and (7), we get

$$\mathcal{A}(\ell) \le \exp\left[\log n - (\ell-1)\Upsilon\left(\frac{\log n}{\ell-1}\right) + \ell \log f(2 - \eta\log n/\ell)\right]\,. \tag{8}$$

Let $c = \ell/\log n$ and $c_1 = c - 1/\log n$. By Lemma 8 and since the function $c\Upsilon(1/c)$ is uniformly continuous in $c \in [0, 2e]$, we find that for large enough n,

$$1 - c_1\Upsilon(1/c_1) + c\log f(2 - \eta/c) < \frac{1}{2}\,\eta(1-p)\log(1-p^3) = -\Omega(1)\,.$$

Together with (8), this gives $\mathcal{A}(\ell) = \exp(-\Omega(\log n))$, and (6) follows. ◀

4 Transformations of the Random-surfer Tree Model

In Sections 4–6 we study the random-surfer tree model and we prove Theorems 3 and 4. In this section we show how to transform the random-surfer tree model three times to eventually obtain a new random tree model, which we analyse in subsequent sections. The first transformation is novel. The second one was used by Broutin and Devroye [7], and the third one by Pittel [22].

Let us call the random-surfer tree model the *first model*. First, we will replace the attachment rule with a simpler one by introducing *weights* for the edges. In the first model, the edges are unweighted and in every step s a new vertex v_s appears, chooses an old vertex u, and attaches to a vertex in the path connecting u to the root, according to some rule. We introduce a second model that is weighted, and such that there is a one to one correspondence

between the vertices in the second model and in the first model. For a vertex v in the first model, we denote its corresponding vertex in the second model by $\overline{v}$. In the second model, in every step s a new vertex $\overline{v_s}$ appears, chooses an old vertex $\overline{u}$ and attaches to $\overline{u}$, and the weight $w(\overline{u}\,\overline{v_s})$ of the new edge $\overline{u}\,\overline{v_s}$ is chosen such that the *weight* of $\overline{v_s}$ equals the *depth* of v_s in the first model. Let $w(\overline{u})$ denote the weight of vertex $\overline{u}$. Then it follows from the definition of the random-surfer tree model that $w(\overline{u}\,\overline{v_s})$ is distributed as $\max\{1-\mathrm{Geo}(p), 1-w(\overline{u})\}$. The term $\mathrm{Geo}(p)$ here precisely corresponds to the length of the random walk corresponding to v_s in Definition 1, and the term $1-w(\overline{u})$ appears here solely because the weight of $\overline{v_s}$ is at least 1 (in the first model, the depth of v_s is at least 1, since it cannot attach to a vertex higher than the root). Let us emphasize that $w(\overline{u}\,\overline{v_s})$ can be negative. Because the depth of v in the first model equals the weight of $\overline{v}$ in the second model, the height of the first model equals the weighted height of the second model.

We will need to make the degrees of the tree bounded, so we define a third model. In this model, the new vertex can attach just to the leaves. In step s a new vertex v_s appears, chooses a random leaf u and attaches to u using an edge with weight distributed as $\max\{1-\mathrm{Geo}(p), 1-w(\overline{u})\}$. Simultaneously, a new vertex u' appears and attaches to u using an edge with weight 0. Then we have $w(u)=w(u')$ and henceforth u' plays the role of u, i. e. the next vertex wanting to attach to u, but cannot do so because u is no longer a leaf, may attach to u' instead. Clearly there exists a coupling between the second and third models in which the weighted height of the third model, when it has $2n-1$ vertices, equals the weighted height of the second model with n vertices. In fact the second model may be obtained from the third one by contracting all zero-weight edges. We can thus study the weighted height of the first model by studying it in the third model.

All the above models were defined using discrete time steps. We now define a fourth model using the following continuous time branching process. At time 0 the root is born. From this moment onwards, whenever a new vertex v is born (say at time κ), it waits for a random time E, which is distributed exponentially with mean 1, and after time E has passed (namely, at absolute time $\kappa+E$) gives birth to two children v_1 and v_2. The weights of the edges vv_1 and vv_2 are generated as follows: vertex v chooses $i\in\{1,2\}$ independently and uniformly at random. The weight of vv_i is distributed as $\max\{1-\mathrm{Geo}(p), 1-w(v)\}$ and the weight of vv_{3-i} is 0. All vertices live forever, and each vertex gives birth to exactly two children during its lifetime. Given $t\geq 0$, we denote by T_t the (almost surely finite) random tree obtained by taking a snapshot of this process at time t. By the memorylessness of the exponential distribution, if one starts looking at this process at any deterministic moment, the next leaf to give birth is chosen uniformly at random. Hence for any stopping time τ, the distribution of T_τ, conditional on T_τ having $2n-1$ vertices, is the same as the distribution of the third model when it has $2n-1$ vertices.

The following lemma implies that certain results for T_t carry over to results for the random-surfer tree model. The proof is by a coupling argument and uses [7, Proposition 2].

▶ **Lemma 9.** *Assume that there exist constants θ_L, θ_U such that for every fixed $\varepsilon>0$,*

$$\mathbb{P}\left[\theta_L(1-\varepsilon)t \leq \mathrm{wht}(T_t) \leq \theta_U(1+\varepsilon)t\right] \to 1$$

as $t\to\infty$. Then for every fixed $\varepsilon>0$, a.a.s. as $n\to\infty$ the height of the random-surfer tree model is between $\theta_L(1-\varepsilon)\log n$ and $\theta_U(1+\varepsilon)\log n$.

We define T_t in a static way, which is equivalent to the dynamic definition above.

▶ **Definition 10** (T_∞, T_t)**.** Let T_∞ denote an infinite binary tree. To every edge e is associated a random pair (E_e, W_e) and to every vertex v a random variable W_v, where the W_e's and

W_v's are the *weights*. The law for $\{E_e\}_{e\in E(T)}$ is easy: first with every vertex v we associate independently an exponential random variable with mean 1, and we let the values of E on the edges joining v to its two children be equal to this variable. In the dynamic interpretation, this random variable denotes the age of v when it gives birth. Generation of the weights is done in a top-down manner, where we think of the root as the top vertex. Let the weight of the root be zero. Let v be a vertex whose weight has been determined, and let v_1, v_2 be its two children. Choose $i \in \{1,2\}$ independently and uniformly at random, and then choose $Y = 1 - \mathrm{Geo}(p)$ independently of previous choices. Then let

$$W_{vv_i} = \max\{Y, 1 - W_v\}, \quad W_{v_i} = W_v + W_{vv_i}\,, \tag{9}$$

and $W_{vv_j} = 0, W_{v_j} = W_v$ for $j = 3 - i$.

For a vertex v, let $\pi(v)$ be the set of edges of the path connecting v to the root. It is easy to check that the weight of any vertex v equals $\sum_{e\in\pi(v)} W_e$. Finally, given $t \geq 0$ we define T_t as the subtree of T_∞ induced by vertices with birth time at most t. Note that T_t is connected by definition, and is finite almost surely.

5 Lower Bounds for the Random-surfer Tree Model

Here we prove the lower bounds in Theorems 3 and 4. For this, we consider another infinite binary tree T'_∞ which is very similar to T_∞, except for the generation rules for the weights, which are as follows. Let the weight of the root be zero. Let v be a vertex whose weight has been determined, and let v_1, v_2 be its two children. Choose $i \in \{1,2\}$ independently and uniformly at random, and choose $Y = 1 - \mathrm{Geo}(p)$ independently of previous choices. Then let

$$W_{vv_i} = Y \text{ and } W_{v_i} = W_v + W_{vv_i} \tag{10}$$

and $W_{vv_j} = 0$ and $W_{v_j} = W_v$ for $j = 3 - i$. Comparing (10) with (9), we find that the weight of every vertex in T'_∞ is stochastically less than or equal to that of its corresponding vertex in T_∞. The tree T'_t is defined as before. Clearly probabilistic lower bounds for $\mathrm{wht}(T'_t)$ are also probabilistic lower bounds for $\mathrm{wht}(T_t)$.

The proof of the following lemma is similar to that of [7, Lemma 4] except a small twist is needed at the end to handle the negative weights. Distinct vertices u and v in a tree are called *antipodal* if the unique (u,v)-path in the tree passes through the root.

▶ **Lemma 11.** *Consider the tree T'_∞. Let $\gamma_L : (0,1) \to \mathbb{R}$ be such that for every $a \in (0,1)$, each vertex u and each descendent v of u that is m levels deeper,*

$$\mathbb{P}\left[W_v - W_u \geq am\right] \geq \exp(-m\gamma_L(a) + o(m)) \tag{11}$$

as $m \to \infty$. Assume that there exist $\alpha^*, \rho^* \in (0,1)$ with $\gamma_L(\alpha^*) + \Upsilon(\rho^*) = \log 2$, where Υ is defined in (3). Then for every fixed $\varepsilon > 0$, a.a.s. there exist antipodal vertices u, v of T'_t with weights at least $\frac{a^*}{\rho^*}(1-\varepsilon)t$.

Proof. Let $c = \frac{a^*}{\rho^*}$, and let $\varepsilon, \delta > 0$ be arbitrary. We prove that with probability at least $1 - \delta$ for all large enough t there exists a pair (u,v) of antipodal vertices of T'_∞ with $\max\{B_u, B_v\} < t$ and $\min\{W_u, W_v\} > (1 - 2\varepsilon)\,ct$.

Let L be a constant positive integer that will be determined later, and let $\alpha = \alpha^*$ and $\rho = \frac{\alpha}{c(1-\epsilon)} > \rho^*$. Since $\gamma_L(\alpha^*) + \Upsilon(\rho^*) = \log 2$ and $\rho^* < 1$ and Υ is strictly decreasing on $(0,1]$, we have

$$\gamma_L(\alpha) + \Upsilon(\rho) < \log 2\,.$$

Build a Galton-Watson process from T'_∞ whose particles are a subset of vertices of T'_∞, as follows. Start with the root as the initial particle of the process. If a given vertex u is a particle of the process, then its potential offspring are its 2^L descendants that are L levels deeper. Moreover, such a descendent v is an offspring of u if and only if $W_v - W_u \geq \alpha L$ and $B_v - B_u \leq \rho L$. As these two events are independent, the expected number of children of u is at least

$$2^L \mathbb{P}[W_v - W_u \geq \alpha L]\, \mathbb{P}[B_v - B_u \leq \rho L] \geq \exp[(\log 2 - \gamma_L(\alpha) - \Upsilon(\rho) - o(1))L]$$

as $L \to \infty$, by (11) and Proposition 7. Since we have $\log 2 - \gamma_L(\alpha) - \Upsilon(\rho) > 0$, we may choose L large enough that this expected value is strictly greater than 1. Therefore, this Galton-Watson process survives with probability $q > 0$.

We now boost this probability up to $1 - \delta$, by starting several independent processes, giving more chance that at least one of them survives. Specifically, let b be a constant large enough that

$$(1-q)^{2^{b-1}} < \delta/3\,.$$

Consider 2^b Galton-Watson processes, which have the vertices at depth b of T'_∞ as their initial particles, and reproduce using the same rule as before. Let a be a constant large enough that

$$2^{b+1}(e^{-a} + (1-p)^{a+2}) < \delta/3\,,$$

and let A be the event that all edges e in the top b levels of T'_∞ have $E_e \leq a$ and $W_e \geq -a$. Then

$$1 - \mathbb{P}[A] \leq 2^{b+1}(e^{-a} + (1-p)^{a+2}) < \delta/3\,.$$

Also, let Q be the event that in each of the two branches of the root, at least one of the 2^{b-1} Galton-Watson processes survives. Then

$$1 - \mathbb{P}[Q] \leq 2(1-q)^{2^{b-1}} < 2\delta/3\,,$$

and so with probability at least $1 - \delta$ both A and Q occur.

Assume that both A and Q occur. Let

$$m = \left\lfloor \frac{t(1-\varepsilon)}{\rho L} \right\rfloor$$

and let u and v be particles at generation m of surviving processes in distinct branches of the root. Then u and v are antipodal,

$$\max\{B_u, B_v\} \leq ab + m\rho L \leq t(1-\varepsilon) + O(1) < t\,,$$

and

$$\min\{W_u, W_v\} \geq -ab + m\alpha L \geq \frac{(1-\varepsilon)\alpha}{\rho}\, t - O(1) > c(1-2\varepsilon)t$$

for t large enough, as required. ◀

Let $Y_1, Y_2, \ldots$ be i.i.d. with $Y_i = 1 - \mathrm{Geo}(p)$. Let $f : (-\infty, 1] \to \mathbb{R}$ be as defined in (4). Note that $f(1) = p$ and $f\left(2 - p^{-1}\right) = 1$. It is easy to verify that f is continuous in $(-\infty, 1]$ and differentiable in $(-\infty, 1)$. Moreover, f is increasing on $(-\infty, 2 - p^{-1}]$ and decreasing on $[2 - p^{-1}, 1]$. Using standard concentration methods, we can show that as $m \to \infty$, uniformly for all $a \in [0, 1]$,

$$\mathbb{P}[Y_1 + \cdots + Y_m \geq am] \geq [f(a) - o(1)]^m\,, \tag{12}$$

and if $p \geq 1/2$, then uniformly for all $a \in [0, 2 - \frac{1}{p}]$ we have

$$\mathbb{P}\left[Y_1 + \cdots + Y_m \geq am\right] \geq [1 - o(1)]^m \ . \tag{13}$$

We define a two variable function $\Phi(a, s) = p(1-p)(2-s)^2(s-a) - a(1-s)$, and we define a function $\phi : [0,1] \to [0,1]$ implicitly by the equation

$$\Phi(a, \phi(a)) = p(1-p)(2-\phi(a))^2(\phi(a)-a) - a(1-\phi(a)) = 0\,. \tag{14}$$

It is not hard to see that ϕ is well defined, increasing and invertible on $[0,1]$, and differentiable on $(0,1)$. Moreover, if $a \in \{0,1\}$ then $\phi(a) = a$, and otherwise, $0 < a < \phi(a) < 1$.

Next let $\hat{Y}_1, \hat{Y}_2, \ldots$ be independent and distributed as follows: for every $i = 1, 2, \ldots$ we flip an unbiased coin, if it comes up heads, then $\hat{Y}_i = Y_i$, otherwise $\hat{Y}_i = 0$. Define $g_L : (0,1) \to \mathbb{R}$ as

$$g_L(a) = \begin{cases} 1/2 & \text{if } p > 1/2 \text{ and } 0 < a < 1 - \frac{1}{2p} \\ \frac{\phi(a)-a}{\phi(a)} \left(\frac{(1-p)(2-\phi(a))}{1-\phi(a)} \right)^a & \text{otherwise.} \end{cases}$$

The inequalities (12) and (13) together with Stirling's formula imply the following.

▶ **Lemma 12.** *We have the following large deviation inequality for every fixed $a \in (0,1)$ as $m \to \infty$.*

$$\mathbb{P}\left[\hat{Y}_1 + \cdots + \hat{Y}_m \geq am\right] \geq (2g_L(a) - o(1))^{-m}\,.$$

The lower bound in Theorem 3 is obtained easily from the following lemma and Lemma 9.

▶ **Lemma 13.** *Given $\varepsilon > 0$, a.a.s as $t \to \infty$ there exist two antipodal vertices u, v of T'_t with weights at least $c_L(p)(1-\varepsilon)t$. In particular, a.a.s. the weighted height of T'_t is at least $c_L(p)(1-\varepsilon)t$.*

Proof. It is easy to see that there is a unique solution $s \in (0,1)$ to $(1-p)(2-s) = \exp(1/s)(1-s)$, and if $p > 1/2$ then $s > 2 - 1/p$. By definition, $c_L = c_L(p) = \exp(1/s)s(2-s)p$. By Lemma 12, the assumption (11) of Lemma 11 holds for $\gamma_L(a) = \log(2g_L(a))$. Let $a = \phi^{-1}(s)$ and let $\rho = 1 - \frac{a}{s}$. Since $s \in (0,1)$ we have $0 < a < s < 1$ and thus $\rho \in (0,1)$ as well. Moreover, since $\Phi(a,s) = 0$, we have $c_L = a/\rho$. It is easy to verify that $\Phi(1 - \frac{1}{2p}, 2 - \frac{1}{p}) = 0$. If $p > 1/2$, then we have $s > 2 - \frac{1}{p}$. Since ϕ^{-1} is increasing, we have $a = \phi^{-1}(s) \geq 1 - \frac{1}{2p}$. This implies that $g_L(a) = \frac{s-a}{s}\exp(a/s)$, hence $\log(2g_L(a)) + \rho - 1 - \log(\rho) = \log 2$, and Lemma 11 completes the proof. ◀

Proof of the lower bound in Theorem 4. Fix $\varepsilon > 0$. Let us define the *semi-diameter* of a tree as the maximum weighted distance between any two antipodal vertices. Clearly, semi-diameter is a lower bound for the diameter, so we just need to show a.a.s. as $n \to \infty$ the semi-diameter of the random-surfer tree with n vertices is at least $(2c_L(p) - \varepsilon)\log n$. By Lemma 13, a.a.s as $t \to \infty$ the semi-diameter of T'_t is at least $(2c_L(p) - \varepsilon)t$. Using an argument similar to the proof of Lemma 9 we may conclude that a.a.s. as $n \to \infty$ the semi-diameter of the third model (of Section 4) with $2n-1$ vertices is at least $(2c_L(p) - \varepsilon)\log n$. Then it is easy to observe that the same is true for the random-surfer tree with n vertices, and the proof is complete. ◀

6 Upper Bounds for the Random-surfer Tree Model

In this section we prove the upper bounds in Theorems 3 and 4. We start with a lemma which complements Lemma 11.

▶ **Lemma 14.** *Let $\gamma_U : [0,1] \to [0,\infty)$ be a continuous function such that for every fixed $a \in [0,1]$ and every vertex v of T_∞ at depth m,*

$$\mathbb{P}\left[\sum_{e\in\pi(v)} W_e > am\right] \le \exp(-m\gamma_U(a) + o(m)) \tag{15}$$

as $m \to \infty$. Define

$$\theta = \sup\left\{\frac{a}{\rho} : \gamma_U(a) + \Upsilon(\rho) = \log 2 : a \in [0,1], \rho \in (0,\infty)\right\}.$$

Then for every fixed $\varepsilon > 0$ we have $\mathbb{P}\left[\mathrm{wht}(T_t) > \theta(1+\varepsilon)t\right] \to 0$ as $t \to \infty$.

The proof of Lemma 14 is similar to that of [7, Lemma 3], in which the assumption (15) is not needed. In fact, in the model studied in [7], the weights $\{W_e : e \in \pi(v)\}$ are mutually independent, and the authors use Cramér's Theorem to obtain a large deviation inequality for $\sum_{e\in\pi(v)} W_e$.

Let $Y_1, Y_2, \ldots$ be i.i.d. with $Y_i = 1 - \mathrm{Geo}(p)$, and define random variables $X_1, X_2, \ldots$ as follows: $X_1 = \max\{Y_1, 1\}$, and for $i \ge 1$, $X_{i+1} = \max\{Y_{i+1}, 1 - (X_1 + \cdots + X_i)\}$. Define $h : [0,1] \to \mathbb{R}$ as

$$h(x) = \begin{cases} 1 & \text{if } p \ge \frac{1}{2} \text{ and } 0 \le x \le 2 - \frac{1}{p} \\ \left(\frac{p}{1-p}\right)^x & \text{if } p < \frac{1}{2} \text{ and } 0 \le x \le \frac{1-2p}{1-p} \\ f(x) & \text{otherwise}. \end{cases}$$

A careful but straightforward analysis using exact formulae gives that there exists an absolute constant C such that for every $a \in [0,1]$ and every positive integer m we have

$$\mathbb{P}\left[X_1 + \cdots + X_m > am\right] \le Cm^2 h(a)^m. \tag{16}$$

Next define random variables $\hat{X}_1, \hat{X}_2, \ldots$ as follows: for every $i = 1, 2, \ldots$ we flip an independent unbiased coin, if it comes up heads, then $\hat{X}_i = X_i$, otherwise $\hat{X}_i = 0$. Define $g_U : [0,1] \to \mathbb{R}$ as

$$g_U(a) = \begin{cases} 1/2 & \text{if } p \ge 1/2 \text{ and } 0 \le a \le 1 - 1/2p \\ (\frac{1-p}{p})^a/2 & \text{if } p < 1/2 \text{ and } 0 \le a \le \frac{1-2p}{2-2p} \\ 1/p & \text{if } a = 1 \\ \frac{\phi(a)-a}{\phi(a)}\left(\frac{(1-p)(2-\phi(a))}{1-\phi(a)}\right)^a & \text{otherwise}, \end{cases}$$

where ϕ is defined by (14). Recall that we have $0 < a < \phi(a) < 1$ for $a \in (0,1)$, so g_U is well defined. The following lemma can be proved using (16), Stirling's formula and standard arguments.

▶ **Lemma 15.** *We have the following large deviation inequality for every $a \in [0,1]$ and every positive integer m, where C' is an absolute constant: $\mathbb{P}\left[\hat{X}_1 + \cdots + \hat{X}_m > am\right] \le C'm^3(2g_U(a))^{-m}$.*

We are ready to prove the upper bound in Theorem 3. The upper bound in Theorem 4 follows immediately as in every tree the diameter is at most twice the height.

Proof of the upper bound in Theorem 3. Let $c_U = c_U(p)$. By Lemma 9 we just need to show that given $\varepsilon > 0$, a.a.s as $t \to \infty$ the weighted height of T_t is at most $(1+\varepsilon)c_U t$. For proving this we use Lemma 14. Lemma 15 implies that condition (15) of Lemma 14 holds with $\gamma_U(a) = \log(2g_U(a))$, so we need only show that

$$c_U = \sup\left\{\frac{a}{\rho} : \log(g_U(a)) + \rho - 1 - \log(\rho) = 0 : a \in [0,1], \rho \in (0,\infty)\right\} \tag{17}$$

Observe that if $0 < g_U(a) \le 1$, there is a unique $\rho \in (0,1]$ satisfying $\log(g_U(a)) + \rho - 1 - \log(\rho) = 0$; and if $g_U(a) > 1$, there is no $\rho \in (0,\infty)$ satisfying this equation. Let $a_{\max} \in [0,1]$ denote the unique solution to $g_U(x) = 1$. Define the function $\tau : [0, a_{\max}] \to (0,1]$ as follows. Let $\tau(a_{\max}) = 1$ and for $x < a_{\max}$ let $\tau(x)$ be the unique number satisfying

$$\log(g_U(x)) + \tau(x) - 1 - \log\tau(x) = 0\,. \tag{18}$$

Hence to prove (17) it is enough to show that

$$c_U = \sup\left\{\frac{x}{\tau(x)} : x \in [0, a_{\max}]\right\}\,. \tag{19}$$

Differentiating (18) and using the implicit function theorem, we find that τ is differentiable and

$$\tau'(x) = \frac{\tau(x)}{1-\tau(x)} \times \frac{g_U'(x)}{g_U(x)}\,.$$

It can be verified that $\log(g_U(a))$ is increasing and convex and that $\rho - 1 - \log(\rho)$ is decreasing and convex. Using standard convexity arguments it can be proved that the supremum in (19) occurs at a point $x^* \in (0, a_{\max})$ with $\tau(x^*) = x^*\tau'(x^*)$, hence it is enough to find $x^* \in (0, a_{\max})$ satisfying

$$c_U = \frac{x^*}{\tau(x^*)} = \frac{1-\tau(x^*)}{\tau(x^*)}\,\frac{g_U(x^*)}{g_U'(x^*)}\,. \tag{20}$$

Recall that $p_0 \approx 0.206$ is the unique solution to (2). If $0 < p \le p_0$, we let $x^* = \left[2\log\left(\frac{1-p}{p}\right)\right]^{-1}$. Then it is not hard to see that

$$g_U(x^*) = \exp\left(\frac{1}{2} - \log 2\right), \quad g_U'(x^*) = \log\left(\frac{1-p}{p}\right) g_U(x^*), \text{ and } \quad \tau(x^*) = 1/2\,,$$

and (20) follows. If $p_0 < p < 1$, we let $x^* = \phi^{-1}(s^*)$, where $s^* \in (0,1)$ is the unique solution to (1). Then (20) follows from the following equations, which are not hard to verify:

$$g_U(x^*) = \left(1 - \frac{x^*}{s^*}\right)\exp\left(x^*/s^*\right), \quad g_U'(x^*) = g_U(x^*)/s^*, \text{ and } \quad \tau(x^*) = 1 - \frac{x^*}{s^*}\,. \qquad \blacktriangleleft$$

7 Concluding Remarks

One natural open problem is to close the gap between the lower and upper bounds in Theorems 3 and 4 when $p < p_0$. It seems that for solving this problem new ideas are required. In Theorem 2 we gave logarithmic upper bounds for the diameter of the random-surfer

Webgraph model for all $d \geq 1$, which are close to being tight when $d = 1$. Another interesting open problem is to give lower bounds for $d > 1$. This problem seems to need quite different techniques. In fact, the diameter for $d > 1$ might be of a smaller order, e. g. $\Theta(\log n / \log \log n)$, as is the case for the preferential attachment model (see [5, Theorem 1]).

There is a common generalization of random recursive trees, preferential attachment trees, and random-surfer trees. Consider i.i.d. random variables $X_1, X_2, \ldots \in \{0, 1, 2, \ldots\}$. Start with a single vertex v_0. At each step s a new vertex v_s appears, chooses a random vertex u in the present graph, and then walks X_s steps from u towards v_0, attaching to the last vertex in the walk (if it reaches v_0 before X_s steps, it attaches to v_0). Random recursive trees correspond to $X_i = 0$, preferential attachment trees correspond to $X_i = \text{Bernoulli}(1/2)$ (see, e. g., [4, Theorem 3.1]), and random-surfer trees correspond to $X_i = \text{Geo}(p)$. Using the ideas of this paper, it is possible to obtain lower and upper bounds for the height and the diameter of this general model (similar to Theorems 3 and 4), provided one can prove large deviation inequalities (similar to (12) and (13)) for the sum of X_i's and also large deviation inequalities (similar to (16)) for the sum of random variables X'_i defined as $X'_1 = 1$ and $X'_{i+1} = \max\{1 - X_i, 1 - (X'_1 + \cdots + X'_i)\}$ for $i > 0$.

References

1 A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

2 S. Bhamidi. Universal techniques to analyze preferential attachment trees: global and local analysis. preprint, available via `http://www.unc.edu/~bhamidi/`, 2007.

3 G. Bianconi and A.-L. Barabási. Competition and multiscaling in evolving networks. *Europhys. Lett.*, 54(4):436–442, 2001.

4 A. Blum, T.-H. H. Chan, and M. R. Rwebangira. A random-surfer web-graph model. In *Proc. of 8th Workshop on Algorithm Engineering and Experiments and 3rd Workshop on Analytic Algorithmics and Combinatorics*, pages 238–246, 2006.

5 B. Bollobás and O. Riordan. The diameter of a scale-free random graph. *Combinatorica*, 24(1):5–34, January 2004.

6 A. Bonato. *A course on the web graph*, volume 89 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2008.

7 N. Broutin and L. Devroye. Large deviations for the weighted height of an extended class of trees. *Algorithmica*, 46(3-4):271–297, 2006.

8 D. Chakrabarti and C. Faloutsos. *Graph Mining: Laws, Tools, and Case Studies*. Synthesis Lectures on Data Mining and Knowledge Discovery. Morgan & Claypool Publishers, 2012.

9 P. Chebolu and P. Melsted. Pagerank and the random surfer model. In *Proceedings of the 19th annual ACM-SIAM symposium on Discrete algorithms*, SODA'08, pages 1010–1018, Philadelphia, PA, USA, 2008.

10 F. Chung and L. Lu. *Complex graphs and networks*, volume 107 of *CBMS Regional Conference Series in Mathematics*. Published for the Conference Board of the Mathematical Sciences, Washington, DC, 2006.

11 A. Dembo and O. Zeitouni. *Large deviations techniques and applications*, volume 38 of *Stochastic Modelling and Applied Probability*. Springer-Verlag, Berlin, 2010. Corrected reprint of the second (1998) edition.

12 L. Devroye, O. Fawzi, and N. Fraiman. Depth properties of scaled attachment random recursive trees. *Random Structures Algorithms*, 41(1):66–98, 2012.

13 S. Dommers, R. van der Hofstad, and G. Hooghiemstra. Diameters in preferential attachment models. *Journal of Statistical Physics*, 139(1):72–107, 2010.

14 E. Drinea, A. Frieze, and M. Mitzenmacher. Balls and bins models with feedback. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA'02, pages 308–315, Philadelphia, PA, USA, 2002.

15 R. Durrett. *Random graph dynamics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge, 2010.

16 G. Ergün and G.J. Rodgers. Growing random networks with fitness. *Physica A: Statistical Mechanics and its Applications*, 303(1–2):261–272, 2002.

17 P. L. Krapivsky and S. Redner. Organization of growing random networks. *Phys. Rev. E*, 63:066123, May 2001.

18 J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1), March 2007.

19 L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, 1999.

20 G. Pandurangan, P. Raghavan, and E. Upfal. Using pagerank to characterize web structure. In *Proceedings of the 8th Annual International Conference on Computing and Combinatorics*, COCOON'02, pages 330–339, London, UK, UK, 2002.

21 G. Pandurangan, P. Raghavan, and E. Upfal. Using pagerank to characterize web structure. *Internet Mathematics*, 3(1):1–20, 2006.

22 B. Pittel. Note on the heights of random recursive trees and random m-ary search trees. *Random Structures and Algorithms*, 5(2):337–347, 1994.

Deterministic Coupon Collection and Better Strong Dispersers*

Raghu Meka[1], Omer Reingold[1], and Yuan Zhou[2]

1 **Microsoft Research Silicon Valley**
{meka,omer.reingold}@microsoft.com
2 **Computer Science Department, Carnegie Mellon University**
yuanzhou@cs.cmu.edu

Abstract

Hashing is one of the main techniques in data processing and algorithm design for very large data sets. While random hash functions satisfy most desirable properties, it is often too expensive to store a fully random hash function. Motivated by this, much attention has been given to designing *s*mall families of hash functions suitable for various applications. In this work, we study the question of designing space-efficient hash families $\mathcal{H} = \{h : [U] \to [N]\}$ with the natural property of *c*overing: $\mathcal{H}$ is said to be covering if any set of $\Omega(N \log N)$ distinct items from the universe (the *c*oupon-collector limit) are hashed to cover all N bins by most hash functions in $\mathcal{H}$. We give an explicit family $\mathcal{H}$ of size $\text{poly}(N)$ (which is optimal), so that hash functions in $\mathcal{H}$ can be specified efficiently by $O(\log N)$ bits.

We build covering hash functions by drawing a connection to *dispersers*, which are quite well studied and have a variety of applications themselves. We in fact need *strong dispersers* and we give new constructions of strong dispersers which may be of independent interest. Specifically, we construct strong dispersers with optimal entropy loss in the high min-entropy, but very small error ($\text{poly}(n)/2^n$ for n bit sources) regimes. We also provide a strong disperser construction with constant error but for any min-entropy. Our constructions achieve these by using part of the source to replace seed from previous non-strong constructions in surprising ways. In doing so, we take two of the few constructions of dispersers with parameters better than known extractors and make them strong.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Coupon collection; dispersers, strong dispersers, hashing, pseudorandomness

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.872

1 Introduction

Hashing is one of the main techniques in data processing and algorithm design for handling large data sets. When processing data from a universe U, to avoid various computational bottlenecks such as storage and load distribution it is often helpful to hash down to a smaller universe (aka bins), via a hash family $\mathcal{H} = \{h : h : U \to [N]\}$. Different applications call for different requirements of the hash family $\mathcal{H}$ and there is a rich body of work on constructions of families with various properties starting with the seminal work of Carter and Wegman [6].

One such prominently studied property is *load-balancing* where one desires no bin to receive too many items. In this paper, we consider the related property of *covering*. Besides

* Work done while the third author was visiting Microsoft Research Silicon Valley.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 872–884

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

being a natural and desirable property by itself, the *covering* property is also useful in achieving load balancing. A classical question in probability theory is the *coupon collector problem*. Suppose that there are N different coupons and at each trial you get one random coupon. How many trials do you need in expectation before you collect all N coupons? The answer of course is $\Theta(N \log N)$. An implication of this is that if we randomly hash $\Omega(N \log N)$ distinct objects from a universe to N bins, then with high probability, the objects cover all the bins. This motivates the following definition formulated by Alon et al. [2][1].

▶ **Definition 1.** A family of hash functions $\mathcal{H} = \{h : U \to [N]\}$ is ϵ-covering if there exists a constant C such that for all subsets $S \subseteq U$, $|S| \geq CN \log N$,

$$\Pr_{h \in_u \mathcal{H}}[h(S) = [N]] \geq 1 - \epsilon.$$

We say $\mathcal{H}$ is covering if it is $1/2$-covering.

The coupon collector argument shows that fully random functions satisfy ϵ-covering property with $\epsilon = N^{-\Omega(1)}$. However, fully random hash functions are inefficient in practice as we need space $|U|$ to describe them and space is critical in many scenarios where hashing is helpful. We address the question of designing efficient hash families with covering property as above which are small or equivalently can be sampled with, and hence described by, few bits. As the covering property of fully random hash functions follows from the coupon collection problem, one can intuitively view our end goal as a *derandomization* of the classical coupon collection process.

Standard families like $O(\log N)$-wise independent hash functions (see preliminaries for formal definition) which are known to have strong load-balancing properties are also $N^{-\Omega(1)}$-covering. However, such hash families have size $N^{O(\log N)}$. Similar parameters were achieved by Alon et al. [2] by using random linear hash functions. The work of Celis et al. [7] gives efficient covering hash families of size $N^{O(\log \log N)}$. Here, we solve the problem by giving the first polynomial size, efficient (logarithmic evaluation time[2]) hash family:

▶ **Theorem 2.** *Let $N > 0$ and $c > 0$. Then, there exists an N^{-c}-covering family of hash functions $\mathcal{H} = \{h : U \to [N]\}$ with $|\mathcal{H}| = ((\log |U|) \cdot N)^{O(1)}$. Moreover, each $h \in \mathcal{H}$ can be evaluated in time $O(\log N)$ from a description of length $O(\log N)$.*

Our construction of such hash families relies on a simple connection between covering hash families and the well studied concept of *randomness dispersers*. While the connection itself is not surprising, the strong dispersers we need were not known before and we give new explicit constructions which are interesting by themselves. In the following subsection, we first briefly recall the basic notions of randomness extractors and dispersers, and introduce our new constructions of dispersers. We then describe the relation to covering hash families.

1.1 Extractors and Dispersers

Extractors and dispersers are important combinatorial objects with many applications throughout computer science. Informally, an extractor is a function which takes a *biased* source of randomness with some entropy and outputs ("extracts") a distribution that is close to the uniform distribution. To achieve this, we are allowed to use a few additional random

[1] Throughout, $x \in_u X$ denotes a uniformly random element from a multi-set X.
[2] In the standard unit-cost RAM model.

bits (*seed*), which is necessary. Here, the entropy is quantified by the notion of *min-entropy*: for any random variable X over a universe U, the min-entropy of X is defined by

$$H_\infty(X) = \min_{a \in U} \log\left(\frac{1}{\Pr[X = a]}\right).$$

We quantify closeness between random variables via the standard statistical distance denoted $\Delta(\ ,\ )$.

▶ **Definition 3** (Extractor, [20]). For $k, \epsilon > 0$, a (k, ϵ)-extractor[3] is a function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that for any random variable X over $\{0,1\}^n$ with $H_\infty(X) \geq k$, we have $\Delta(X, U_n) \leq \epsilon$.[4]

We say the extractor is *strong* if the extractor output is close to uniform even given the seed:

$$\Delta((\mathsf{Ext}(X, U_d), U_d), (U_m, U_d)) \leq \epsilon.$$

We refer to the parameter d as the seed-length of the extractor and say the extractor is explicit if the function Ext can be computed in time which is polynomial in n. We refer to $k + d - m$ as the *entropy loss* of the extractor as this corresponds intuitively to the number of random bits we lose in the extraction process; for strong extractors, the entropy loss is defined to be $k - m$.

Closely related to extractors are dispersers, which can be seen as a relaxation of extractors where instead of requiring the output distribution to be close to uniform, we only require the output distribution to have large support.

▶ **Definition 4** (Disperser). For $k, \epsilon > 0$, a (k, ϵ)-disperser is a function $Dsp : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that for any random variable X over $\{0,1\}^n$ with $H_\infty(X) \geq k$, we have $|\mathrm{Supp}(Dsp(X, U_d))| \geq (1 - \epsilon)2^m$.

We say the disperser is *strong* if the output of the disperser has large support for most seeds:

$$|\mathrm{Supp}((Dsp(X, U_d), U_d))| \geq (1 - \epsilon)2^{m+d}.$$

Similar to extractors, we refer to the parameter d as the seed-length of the disperser and say the disperser is explicit if the function Dsp can be computed in time which is polynomial in n. We refer to $k + d - m$ as the entropy loss of the disperser; for strong dispersers, the entropy loss is defined to be $k - m$.

Over the past few decades, extractors and dispersers have been the focus of intense study. In particular, because of their many pseudo-random properties, extractors and dispersers have by now become standard tools in complexity theory and algorithm design with numerous applications: e.g., error-correcting codes (e.g., [27]), cryptography (e.g., [9], [17]) and pseudorandom generators (e.g., [20], [5]). We refer to the recent survey of Vadhan [28], and the references therein for more details.

It can be shown by the probabilistic method that a) there exist strong extractors with seed-length $d = \log(n - k) + 2\log(1/\epsilon) + O(1)$ and entropy loss of $2\log(1/\epsilon) + O(1)$; b) there exist strong dispersers with seed-length $d = \log(n - k) + \log(1/\epsilon) + O(1)$ and entropy loss of $\log\log(1/\epsilon) + O(1)$. These bounds were also shown to be tight up to additive $O(1)$ terms by Radhakrishnan and Ta-Shma [21]. However, most applications of extractors and dispersers require explicit constructions and most effort in the area has been towards giving explicit

[3] We often omit n, m as they will be clear from context.
[4] Throughout, U_n denotes the uniform distribution on $\{0,1\}^n$.

constructions matching the above bounds. Indeed, we do have several strong and nearly optimal constructions in many important parameter regimes (see [14], [10] for some of the most recent constructions). However, we do not yet have the best constructions in many other regimes and this is especially so when dealing with very small errors, which is what we need. Here we address the construction of strong dispersers with very small error and show the following.

▶ **Theorem 5.** *For all $n, 0 \leq \epsilon \leq 1/2, c \geq 1$, there exists an explicit $(n-c,\epsilon)$-strong disperser $Dsp : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(2^c \log(1/\epsilon))$ and entropy loss $(n-c) - m = \log\log(1/\epsilon) + c + O(1)$.*

The main feature of the above construction is the optimal (up to an additive constant) bound on entropy loss for *any* error parameter ϵ, in particular, even for error as small as $2^{-n}/\mathrm{poly}(n)$). Previously, explicit dispersers with optimal entropy loss as above were known [13], but they were not strong. Being able to handle very small errors and having strong dispersers will both be important for the application to covering hash families.

In addition, using a different set of ideas, we also build a strong disperser with small entropy loss for all min-entropies but for constant error ϵ.

▶ **Theorem 6.** *For all n, k, and constant ϵ, there exists an explicit (k,ϵ)-strong disperser $Dsp : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n)$ and entropy loss $k - m = 3\log n + O(1)$.*

While we do not use the above construction here, it follows a similar theme as in Theorem 5 and could be of independent interest.

2 Techniques

Let us first examine the connection between covering hash families and strong dispersers. Let $\mathcal{H} = \{h : U \to [N]\}$ be a ϵ-covering family of hash functions. Let $U \equiv \{0,1\}^n$ and $[N] \equiv \{0,1\}^m$. Then, we can define a function $Dsp : \{0,1\}^n \times \mathcal{H} \to \{0,1\}^m$ by $Dsp(x,h) = h(x)$. Clearly, Dsp can be viewed as a plausible disperser with seed-length $d = \log(|\mathcal{H}|)$ and conversely, any disperser with these parameters defines a corresponding hash family $\mathcal{H}$ from $\{0,1\}^n$ to $\{0,1\}^m$.

By setting up the definitions appropriately, it is not hard to show that ϵ-covering hash families imply (k,ϵ)-strong dispersers for $k = \log_2(N\log N) + O(1) = m + \log_2 m + O(1)$ and $\epsilon = N^{-O(1)}$. Similarly, (k,ϵ)-strong dispersers as above imply ϵ-covering hash families. However, note that the entropy loss of the dispersers we want is

$$k - m = \log_2 m + O(1) = \log\log_2(1/\epsilon) \pm O(1).$$

Therefore, ϵ-covering hash functions necessitate strong dispersers with optimal entropy loss (up to $O(1)$ additive term) and very small error. We achieve such hash functions by appealing to Theorem 5. The actual construction proceeds in two steps.

We first hash the universe U to $[CN\log N]$ bins for a sufficiently large constant C so that the number of distinct bins hit is $\Omega(N\log N)$. We do so by using almost $O(1)$-wise independent hash functions. In the terminology of extractors and dispersers, this step can be seen as *condensing* the source so as to boost the relative min-entropy of the source. In particular, to obtain covering hash families from strong dispersers as outlined in the above argument, we now only need a disperser which works for entropy deficiency at most

$$\log(CN\log N) - \log(\Omega(N\log N)) = O(1),$$

as exactly achieved in Theorem 5. Thus, to get our final covering hash family, we hash from $[CN \log N]$ to $[N]$ bins using the strong disperser from Theorem 5.

We next discuss our constructions of dispersers.

2.1 Strong Dispersers

As remarked earlier, the main problem with using existing constructions for the dispersers we want is that the known constructions are not strong. This difference is not crucial for extractors as most known constructions are either strong or can be made strong via the reduction of Raz, Reingold and Vadhan [23]. No such generic reductions are known for dispersers.

The main insight in our constructions is to use the source to replace part of the seed from the previous non-strong constructions. We will shortly discuss why this is useful. This usually does not work with some notable exceptions being the works of Gabizon, Raz, Shalitel [11] and Barak et al. [3], Barak et al. [4]. Each of our constructions achieve this in a different way and a different analysis.

For the high entropy construction (Theorem 5), we use the techniques of Gradwohl et al. [13] which in turn rely on the classical *expander walk theme*. Roughly speaking, the disperser of Gradwohl et al. is obtained as follows. They first associate the source strings with the vertices of an expander and then compute the output of the disperser by taking a certain walk as specified by the seed on the expander graph. However, their construction implicitly involves a *guess* for how many steps to take in the random walk and this makes their constructions non-strong. We in turn use a part of the source to determine how many steps to take. This causes two problems. Firstly the part of the source we use is not fully random. Secondly, and perhaps more seriously, this also induces probabilistic dependencies between the starting point of the walk and the *edge labels* for the random walk. However, we show that the expander walk we take is robust enough to handle these issues.

For the general entropy disperser, our construction relies on the basic idea of splitting the source into a *block-wise source* which has been used in many constructions of extractors and dispersers. However, most previous constructions *guessed* a good splitting point for the source and this is the reason why they seem inherently non-strong as most of the guesses are usually wrong. Our approach is to first extract $\log n$ bits from the n-bit source and use them to determine the splitting point. We then argue that, despite the subtle dependencies introduced by this step, known constructions of extractors and dispersers for block-wise sources are robust enough to still work.

We give more details of our constructions along with the previous ones we build on in the corresponding sections.

3 Preliminaries

We start with some basic notations and definitions from probability.

- We use U_n to denote the uniform distribution over $\{0,1\}^n$.
- Given a discrete random variable X, we use $\mathrm{Supp}(X)$ to denote the support of X, i.e. the set of elements a such that $\Pr[X = a] > 0$. We call a distribution X *flat* if X is the uniform distribution over $\mathrm{Supp}(X)$.
- The statistical or variational distance between random variables X, Y over a universe U is defined as

$$\Delta(X,Y) = \max_{A \subseteq U} |\Pr[X \in A] - \Pr[Y \in A]| .$$

We say that X, Y are ϵ-close (or X is ϵ-close to Y) if $\Delta(X, Y) \leq \epsilon$.

We shall use the following easy fact.

▶ **Fact 7.** *If X is ϵ-close to U_n, then $|\mathrm{Supp}(X)| \geq (1-\epsilon)2^n$.*

We shall also use the following basic tools from pseudorandomness.

3.1 Expander Graphs

▶ **Definition 8** (Expander Graphs)**.** Let $G = (V, E)$ be a regular graph with normalized adjacency matrix A. We call G a (D, λ)-expander if G is D-regular and the second largest eigenvalue (in absolute value) of A is at most λ. We say G is explicit if there exists an algorithm that, given any vertex $v \in V$, and an index $i \in [D]$, computes the i-th neighbor of v in time $\mathrm{poly}\log|V|$.

Explicit expanders with almost optimal trade-off between degree D and expansion λ are constructed (see, e.g. the work by Lubotzky, Philips and Sarnak [18]). In this paper, we only use the fact that for every constant λ, there exists a constant D and explicit (D, λ)-expanders for every V. Explicit constructions of such expanders where the evaluation can be done with $O(1)$ word operations are known (for example Margulis's expander, see [16]).

We will use the following sampling lemma essentially from [12] (the version we state with better constants and different sets $S_1, \ldots, S_t$ follows easily from [15]).

▶ **Theorem 9.** *Let $G = (V, E)$ be a (D, λ)-expander on V. Consider a random walk $X_0, X_1, \ldots, X_t$ on G, where X_0 is a random start vertex in V. Then, for all $S_1, S_2, \ldots, S_t \subseteq V$ with $\mu = \left(\sum_{i=1}^{t} |S_i|/|V|\right)/t$,*

$$\Pr[\forall i,\ X_i \notin S_i] \leq \exp\left(-\mu^2(1-\lambda)t/4\right).$$

3.2 Hash Functions with Limited Independence

▶ **Definition 10** (k-wise independent Hash Functions)**.** A hash family $\mathcal{H} = \{h : h : U \to [N]\}$ is δ-almost k-wise independent if for all distinct $u_1, \ldots, u_k \in U$, and $h \in_u \mathcal{H}$, $(h(u_1), \ldots, h(u_k))$ is δ-close to the uniform distribution on $[N]^k$. We say the hash family is explicit if the elements of $\mathcal{H}$ can be sampled efficiently.

Using ϵ-biased distributions ([19]) one can design efficient hash families as above of small size.

▶ **Theorem 11** ([19])**.** *For all U, N, there exists an explicit δ-almost k-wise independent hash family $\mathcal{H} = \{h : [U] \to [N]\}$ with $\log(|\mathcal{H}|) = O((\log\log U) + k\log N + \log(1/\delta))$. Further, for a given input, the output of any function in the family can be evaluated in $O(k\log N + \log(1/\delta))$ word operations in the unit cost RAM model.*

3.3 Known Extractors

Our constructions rely on some previous constructions of extractors which we review next.

The following constructions of Ta-Shma, Umans and Zuckerman [26] and Srinivasan and Zuckerman [25] give extractors with nearly optimal entropy losses but sub-optimal seed-lengths.

▶ **Theorem 12** ([26])**.** *For every n, k, and constant ϵ, there exist explicit (k, ϵ)-extractors $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log^2 k(\log\log k)^2)$ and $m = k + d - O(1)$.*

▶ **Theorem 13** ([25]). *For every n, k, ϵ, there exists explicit (k, ϵ)-strong extractors $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(k + \log n + \log(1/\epsilon))$ and $m = k - O(\log(1/\epsilon))$.*

The following theorem gives a way to convert explicit extractors to explicit strong extractors.

▶ **Theorem 14** ([24]). *Any explicit (k, ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ can be transformed into an explicit $(k, O(\sqrt{\epsilon}))$-strong extractor $E' : \{0,1\}^n \times \{0,1\}^{d+d'} \to \{0,1\}^{m-(d+L+1)}$ where $d' = \operatorname{poly}\log(d/\epsilon)$ and $L = 2\log(1/\epsilon) + O(1)$.*

Applying Theorem 14 to Theorem 12, we get the following corollary.

▶ **Corollary 15.** *For every n, k and constant ϵ, there exist explicit (k, ϵ)-strong extractors $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log^2 k(\log\log k)^2)$ and $m = k - O(1)$.*

We shall also need the well-studied notion of block-wise sources [8].

▶ **Definition 16** (block-wise source). Two (possibly correlated) distributions X_1, X_2 form a (k_1, k_2) block-wise source if $H_\infty(X_1) \geq k_1$ and for every x_1, $H_\infty(X_2|X_1 = x_1) \geq k_2$. X is called a (k_1, k_2) block-wise source if $X = X_1 \circ X_2$ where X_1, X_2 form a (k_1, k_2) block-wise source.

▶ **Definition 17** (subsource). A distribution X' over domain $\{0,1\}^n$ is a subsource of a distribution X (over the same domain $\{0,1\}^n$) if there exists an event $A \subset \{0,1\}^n$ such that X' is the probability distribution obtained by conditioning on X being in A.

The following simple lemma says that any source has a subsource which is a block-wise source with high min-entropy.

▶ **Lemma 18.** *Let X be a flat distribution over $\{0,1\}^n$ with $H_\infty(X) \geq k$. For every $k_0 < k$, there exists a subsource $X' = X_1 \circ X_2$ of X which is a $(k - k_0 - \log n - 3, k_0)$ block-wise source.*

Proof. Let X be uniformly random over $A \subseteq \{0,1\}^n$ where $|A| \geq 2^k$. For each $0 \leq i \leq n$ and each $u \in \{0,1\}^i$, let A_u be the set of elements in A whose i-prefix is u. Let

$$S = \{(i,u) : 1 \leq i \leq n, u \in \{0,1\}^i, \text{ and } 2^{k_0} \leq |A_u| < 2^{k_0+1} \leq |A_{u_{1,\ldots,(i-1)}}|\},$$

where $u_{1,\ldots,(i-1)}$ is the $(i-1)$-prefix of u. Also let

$$S_i = \{u : (i,u) \in S\}$$

for each $1 \leq i \leq n$. One can show that $\sum_{(i,u)\in S} |A_u| \geq |A|/2 \geq 2^{k-1}$. Therefore $|S| \geq 2^{k-k_0-2}$. Thus there exists an i such that $|S_i| \geq 2^{k-k_0-\log n-2}$, and by letting $A = \cup_{u\in S_i} A_u$, $X' = (X|A)$ is the desired block-wise subsource. ◀

4 Strong Disperser for High Min-Entropy Sources

We now prove Theorem 5. Our construction builds on the work of Gradwohl et al. [13] and amplifies their construction by using a part of the source as a seed. As remarked in the introduction such approaches often do not work or at the very least are quite subtle. We first briefly review the main constructions of Gradwohl et al.

The starting point for the work of Gradwohl et al. is the classical expander walk theme for constructing pseudorandom objects as used for example in [1], [12]. However, they

introduced a twist of the standard expander walk – the *reversed walk*. Let $Dsp : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^{\log t} \to \{0,1\}^m$ be the disperser we are aiming for. Let G be an expander graph with degree D on $\{0,1\}^n$ and constant spectral gap. Let us also associate the *seeds* $\{0,1\}^d \times \{0,1\}^{\log t}$ with tuples $([D]^t, \ell)$ for $\ell \in [t]$ and suitable parameters D, t. The disperser of [13] takes a ℓ-step walk on the expander with edge-labels as given by the seed, but in the *reverse order*: for $x \in \{0,1\}^n$, $y = (e_1, \ldots, e_t) \in [D]^t$, $\ell \in [t]$, $Dsp(x,y) =$ vertex reached by taking the walk $(e_\ell, e_{\ell-1}, \ldots, e_1)$ in G starting at x.

The idea of taking the walk in reverse order in comparison to the works of [1], [12], which may seem to be of little significance at first look is in fact important for the (simple) analysis of [13]. The final analysis is based on an application of Theorem 9. Unfortunately, the disperser obtained by this approach is not strong and we need some new ideas to make it strong.

Observe that in the above construction, by looking at the ℓ-step random walk for all $\ell \in [t]$, we indirectly allow ourselves to look at a *random* intermediate point of a t-step random walk. This enables [13] to decrease the error probability even further (compared to standard expander walk constructions) to the desired bound. However, doing so costs at least $\log t$ additional random bits which is too much of a loss for us. We instead use a part of the source string itself, say the last $\log t$ bits, to get these additional $\log t$ bits. Note that this introduces problematic (probabilistic) dependencies between the starting point of the walk and the number of steps of the walk. We argue that the expander walk construction is robust enough to handle such dependencies to get our final construction for sources with $O(1)$ entropy deficiency ($(n-k)$).

Proof of Theorem 5. Our simple construction and analysis follow the above sketch closely. We first set up some parameters. Let $t = 2^{2c+3}\log(1/\epsilon)$, $m = n - \log t = n - \log\log(1/\epsilon) - 2c - 3$. Let D be a sufficiently large constant to be chosen later and $G = (\{0,1\}^m, E)$ be an explicit $(D, 1/2)$-expander. Finally, let $d = t \log D$.

Given input $(X, Y) \in \{0,1\}^n \times \{0,1\}^d$, we split X as $(X_0 \circ i)$ where $X_0 \in \{0,1\}^m$, and $i \in \{0,1\}^{\log t}$ which we will view as an integer in $[t]$. We also view the seed Y as $Y \equiv (e_1, \ldots, e_t) \in [D]^t$ in lieu of performing a random walk on G.

Define $Dsp(X,Y)$ as follows. For $\ell \in [t]$, let $X_\ell \in \{0,1\}^m$ be the vertex in G reached by traversing the edges $(e_\ell, e_{\ell-1}, \ldots, e_1)$ (in that order) from the vertex X_0. Then,

$$Dsp((X_0, i), (e_1, \ldots, e_t)) = X_i.$$

We next argue that Dsp as defined above is a $(n - c, \epsilon)$-strong disperser. To see this, fix a set $S \subseteq \{0,1\}^n$ with $|S| \geq 2^{n-c}$. For $y \in \{0,1\}^d$ and $z \in \{0,1\}^m$, call (y,z) *bad* if $z \notin Dsp(S,y)$, i.e., there is no $x \in S$ such that $Dsp(x,y) = z$. We will show that the fraction of bad pairs $(y,z) \in \{0,1\}^d \times \{0,1\}^m$ is at most ϵ.

▶ **Claim 19.** *We have* $|\{(y,z) \in \{0,1\}^d \times \{0,1\}^m : (y,z) \text{ bad}| \leq \epsilon \cdot 2^d \cdot 2^m$.

Proof. Recall that $n = m + \log t$. For each $i \in [t]$, let $S_i \subseteq \{0,1\}^m$ be the first m bits of the strings in S whose last $\log t$ bits equal i:

$$S_i = \{v \in \{0,1\}^m : v \circ i \in S\}.$$

Note that, $\mu := (\sum_i |S_i|)/t2^m \geq |S|/t2^m \geq 2^{-c}$.

Let $(y,z) \in \{0,1\}^d \times \{0,1\}^m$ be bad, where $y = (e_1, \ldots, e_t)$. Then, by the definition of Dsp, it means that the i-step walk $(e_i, e_{i-1}, \ldots, e_1)$ starting from any element of $v \in S_i$ does not end at the vertex $z \in \{0,1\}^m$. This is equivalent to saying that the i-step walk

$(e_1, \ldots, e_i)$ starting at z does not end at an element of S_i for all i. As for $z \in_u \{0,1\}^m$, $y \in_u \{0,1\}^d$ the above corresponds to a t-step random walk in G, by applying Theorem 9 to the walk, we get

$$\begin{aligned}\Pr[(y,z) \text{ bad}] &= \Pr[\forall i, \text{ Starting at } z, Walk(e_1, \ldots, e_i) \text{ does not land in } S_i] \\ &\leq \exp\left(-\mu^2 t/8\right) \\ &\leq \exp\left(-2^{-2c} t/8\right) = \exp(-\log(1/\epsilon)) \leq \epsilon.\end{aligned}$$

◀

To prove the theorem, let $X \subseteq \{0,1\}^n$ be a $(n-c)$ min-entropy source. Then, $|\mathrm{Supp}(X)| \geq 2^{n-c}$. And, by applying the above arguments to $S = \mathrm{Supp}(X)$, we get

$$|\mathrm{Supp}((Dsp(X, U_d), U_d))| = \{(y,z) : z \in Dsp(S,y)\} \geq (1-\epsilon)2^{m+d}.$$

◀

5 Covering Hash Families

We now prove Theorem 2. Recall that the goal is to hash from a universe U to $[N]$ so as to satisfy ϵ-covering property. As described in the introduction, the construction proceeds in two steps. For the first step, we shall use the following simple property of almost $O(1)$-wise independent hash functions.

▶ **Lemma 20.** *For any integer $d > 1$, let $\mathcal{H} = \{h : U \to [N]\}$ be a family of N^{-2d}-almost $2d$-wise independent hash functions. Then for any $S \subseteq U$ such that $|S| \geq N$, we have,*

$$\Pr_{h \in \mathcal{H}}[|h(S)| \leq N/4] \leq O_d\left(\frac{1}{N^{d-1}}\right).$$

Proof. For any set $S \subseteq U$ of cardinality N, let the random variable T be the number of distinct pair (i,j)'s such that $i, j \in S$ and $h(i) = h(j)$, where h is a random element from $\mathcal{H}$. One can show that

$$\mathbf{E}[T^d] \leq O_d(N).$$

Therefore

$$\Pr[T \geq N] \leq \mathbf{E}\left[\frac{T^d}{N^d}\right] = O_d\left(\frac{1}{N^{d-1}}\right).$$

The lemma follows by the fact that when $|h(S)| < N/4$, the T value for h is at least N. ◀

We are now ready to construct the family of hash functions in Theorem 2, proving our main result.

Proof of Theorem 2. We start with the construction of the set of hash functions. We will only work with $N = 2^n$ for some integer $n > 0$. It is easy to extend the construction to general N's. The construction proceeds in two stages.

Let $Dsp : \{0,1\}^{n+\log n+\alpha} \times \{0,1\}^d \to \{0,1\}^n$ be a $(n+\log n+\alpha-2, 1/2^{(\lceil c+2 \rceil)(n+1)})$-strong disperser as constructed in Theorem 5. We know that $\alpha = O(1)$ and $d = O(n)$. Further, as the disperser takes an $O(n)$ walk on an expander, it can be computed with $O(n) = O(\log N)$ word operations.

Let $\mathcal{G} : \{g : U \to [2^\alpha N \log N]\}$ be a family of $(2^\alpha N \log N)^{-2\lceil c+2\rceil}$-almost $2\lceil c+2\rceil$-wise independent functions constructed in Theorem 11. A random hash function $h : U \to [N] \in \mathcal{H}$ is defined as follows. Pick a random $g \in \mathcal{G}$, and pick a random $r \in \{0,1\}^d$. Let

$$h(x) = Dsp(g(x), r).$$

By Theorem 11, any $g \in \mathcal{G}$ can be described with $O((\log\log U) + c\log N)$ bits and computed with $O(\log N)$ word operations. Therefore, we have $|\mathcal{H}| \le ((\log U) \cdot N)^{O(1)}$ and each function in $\mathcal{H}$ can be computed with $O(\log N)$ word operations.

Now we will prove that for any $S \subseteq U$ such that $|S| \ge 2^\alpha N \log N$, the probability (over a random $h \in \mathcal{H}$) that $h(S) \ne [N]$ is at most N^{-c}. By Lemma 20, the probability that $|g(S)| < 2^{\alpha-2} N \log N$ is at most $O(N^{-c-1}) < N^{-c}/2$ (for sufficiently large N). When $|g(S)| \ge 2^{\alpha-2} N \log N$, by the definition of Dsp, with probability at least $(1 - 1/2^{\lceil c\rceil(n+1)}) \ge 1 - N^{-c}/2$ over the random choice of r, we have $|Dsp(g(S), r)| \ge (1 - 1/2^{n+1})2^n$, i.e. $Dsp(g(S), r) = \{0,1\}^n$. By applying a union bound over these two events, we prove the desired statement. ◀

6 Strong Disperser for General Sources

In this section we will prove Theorem 6. We first construct a strong disperser for *block-wise* sources and reduce the general case to that of block-wise sources. The high-level idea is as follows: we first apply an extractor on the second source X_2 to get a short string, which we then use as a seed for applying a strong extractor to the first source X_1. This approach has been used in many other works and is in fact central to many constructions of extractors and dispersers, see [22] for a recent example. We present it here for completeness.

▶ **Lemma 21.** *For all $n, i, k_1 \le i, k_2 \le n - i$, let $X_1 \circ X_2 \in \{0,1\}^i \times \{0,1\}^{n-i}$ be a (k_1, k_2) block-wise source. Suppose that $\log^3 n \le k_2 \le \exp(\log^{1/3} n)$. Then, for all constant $\epsilon > 0$, there exists an explicit strong disperser $Dsp : \{0,1\}^i \times \{0,1\}^{n-i} \times \{0,1\}^d \to \{0,1\}^m$ with error ϵ, where $d = O(\log n)$ and the entropy loss $k_1 + k_2 - m = O(1)$.*

Proof. Let $\delta = \epsilon/2$. Let $E_1 : \{0,1\}^{n-i} \times \{0,1\}^d \to \{0,1\}^{d'}$ be a (k_2, δ)-strong extractor as in Corollary 15, where $d = O(\log n + \log^2 k_2 (\log\log k_2)^2) = O(\log n)$ and $d' = k_2 - O(1)$. Since $\log^3 n \le k_2 \le \exp(\log^{1/3} n)$, $d' = \Omega(\log^3 n)$. Let $E_2 : \{0,1\}^i \times \{0,1\}^{d'} \to \{0,1\}^{m'}$ be a (k_1, δ)-strong extractor in Corollary 15, where $m' = k_1 - O(1)$. This is possible since $d' = \Omega(\log^3 n)$ has more bits than required for Corollary 15 ($O(\log n + \log^2 k_1 (\log\log k_2)^2)$).

Now, we construct our disperser $Dsp : \{0,1\}^i \times \{0,1\}^{n-i} \times \{0,1\}^d \to \{0,1\}^m$, where $m = m' + d' = k_1 + k_2 - O(1)$:

$$Dsp(x_1, x_2, r) = E_2(x_1, E_1(x_2, r)) \circ E_1(x_2, r).$$

Now we show that

$$|\mathrm{Supp}\,(Dsp(X_1, X_2, U_d) \circ U_d)| \ge (1 - 2\delta)2^{m+d},$$

and therefore Dsp is a strong disperser with error at most $2\delta = \epsilon$.

Fix an x_1 from the distribution X_1. Let $Y = E_1(X_2 | X_1 = x_1, r)$ be a distribution, where r is uniformly chosen from U_d. By the definition of block-wise source, we know that $H_\infty(X_2 | X_1 = x_1) \ge k_2$. Therefore, by the definition of strong extractors, (Y, r) is δ-close to $U_{d'+d}$. In particular,

$$\Delta[(E_2(x_1, Y), Y, r), (E_2(x_1, U_{d'}), U_{d'}, U_d)] \le \delta$$

holds for every x_1, where the two copies of $U_{d'}$ denote the same random variable. Therefore, by taking the (weighted) average over x_1, we have

$$\Delta[(E_2(X_1, Y), Y, r), (E_2(X_1, U_{d'}), U_{d'}, U_d)] \leq \delta,$$

where the two copies of $U_{d'}$ denote the same random variable. By the definition of E_2 we have

$$\Delta[(E_2(X_1, U_{d'}), U_{d'}, U_d), (U_{m'+d'}, U_d)] \leq \delta,$$

where the two copies of $U_{d'}$ denote the same random variable. Therefore,

$$\Delta[(E_2(X_1, Y), Y, r), U_{m'+d'+d}] \leq 2\delta.$$

Our claim is proved by observing that $Dsp(X_1, X_2, r) = (E_2(X_1, Y), Y)$, $m = m' + d'$, and Fact 7.

◀

6.1 Proof of Theorem 6

Now we are ready to prove Theorem 6.

Proof of Theorem 6. We will assume that the min-entropy k is at least $\log^4 n$, otherwise the extractor defined in Corollary 15 is good enough to be our disperser.

Let $E : \{0,1\}^n \times \{0,1\}^{O(\log n)} \to [n]$ be an $(\Omega(\log n), \delta/(2n))$-strong extractor as in Theorem 13. For $i \leq n$, let $D_i : \{0,1\}^i \times \{0,1\}^{n-i} \times \{0,1\}^{d'} \to \{0,1\}^m$ be a disperser described in Lemma 21 with $k_1 = k - \log^3 n - 3\log n - 5, k_2 = \log^3 n$. Observe that $d' = O(\log n)$ and $m = k_1 + k_2 - O(1) = k - 3\log n - O(1)$.

Now we define the disperser Dsp as follows. Given input $(x, r) \in \{0,1\}^n \times \{0,1\}^d$, we break r into $r_1 \circ r_2$ where each of r_1 and r_2 has $\Omega(\log n)$ bits. Let $i = E(x, r_1)$. Let $Dsp(x, r) = D_i(x, r_2)$.

We now prove that Dsp is the desired disperser. Without loss of generality, we can assume that the source is a flat distribution, i.e., assume that X is uniform on $A \subseteq \{0,1\}^n$ where $|A| \geq 2^k$.

By the definition of E, with probability at least $(1 - \delta)$ over the random choice of r_1, the distribution $E(X, r_1)$ is $1/(2n)$-close to the uniform distribution over $[n]$. We fix such an r_1 in the following analysis.

By Lemma 18, there exists an i_0 such that there exists $X_1 \circ X_2$ being a $(k - \log^3 n - 2\log n - 4, \log^3 n + \log n + 1)$ subsource of X, where X_1 has the first i_0 bits of the string and X_2 has the remaining $(n - i_0)$ bits. Now, as $i = E(x, r_1)$ is $(1/2n)$-close to being uniform on $[n]$, $\Pr[i = i_0] \geq 1/(2n)$. Let $\tilde{X_1} \circ \tilde{X_2}$ be the random variable obtained by conditioning $X_1 \circ X_2$ on the event $i = i_0$. Then, each of $\tilde{X_1}$, $\tilde{X_2}$ have at most $\log n$ bits less entropy compared to X_1, X_2 (respectively). Therefore, from the above observations and our choice of parameters we get that $\tilde{X_1} \circ \tilde{X_2}$ is a (k_1, k_2) block-wise source.

By Lemma 21, we know that

$$|\mathrm{Supp}(D_{i_0}(\tilde{X_1}, \tilde{X_2}, r_2), r_2)| \geq (1 - \delta)2^{m+|r_2|},$$

therefore

$$|\mathrm{Supp}(D_i(X_1, X_2, r_2), r_2)| \geq (1 - \delta)2^{m+|r_2|}.$$

Since $X_1 \circ X_2$ is a subsource of X, we have

$$|\mathrm{Supp}(D_i(X, r_2), r_2)| \geq (1-\delta)2^{m+|r_2|}.$$

To summarize, we have proved that with probability at least $(1-\delta)$ over the random choice of r_1,

$$|\mathrm{Supp}(D_i(X, r_2), r_2)| \geq (1-\delta)2^{m+|r_2|},$$

which implies that

$$|\mathrm{Supp}(D_i(X, r_2), r_1, r_2)| \geq (1-\delta)2^{m+|r_1|+|r_2|}.$$

Our claim is proved by observing that $Dsp(X, r_1, r_2) = D_i(X, r_2)$. ◀

References

1. Miklós Ajtai, János Komlós, and Endre Szemerédi. Deterministic simulation in LOGSPACE. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 132–140. ACM, 1987.
2. Noga Alon, Martin Dietzfelbinger, Peter Bro Miltersen, Erez Petrank, and Gábor Tardos. Linear hash functions. *J. ACM*, 46(5):667–683, 1999.
3. 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 37th Annual ACM Symposium on Theory of Computing*, pages 1–10, 2005.
4. Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2-source dispersers for sub-polynomial entropy and ramsey graphs beating the frankl-wilson construction. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 671–680, 2006.
5. Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, pages 40–47, 2010.
6. J Lawrence Carter and Mark N Wegman. Universal classes of hash functions. *Journal of computer and system sciences*, 18(2):143–154, 1979.
7. L Elisa Celis, Omer Reingold, Gil Segev, and Udi Wieder. Balls and bins: Smaller hash families and faster evaluation. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, pages 599–608. IEEE, 2011.
8. Benny Chor and Oded Goldreich. On the power of two-point based sampling. *J. Complexity*, 5(1):96–106, 1989.
9. Yevgeniy Dodis, Amit Sahai, and Adam Smith. On perfect and adaptive security in exposure-resilient cryptography. In *EUROCRYPT*, pages 301–324, 2001.
10. Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 181–190, 2009.
11. Ariel Gabizon, Ran Raz, and Ronen Shaltiel. Deterministic extractors for bit-fixing sources by obtaining an independent seed. *SIAM J. Comput.*, 36(4):1072–1094, 2006.
12. Oded Goldreich, Russell Impagliazzo, Leonid A. Levin, Ramarathnam Venkatesan, and David Zuckerman. Security preserving amplification of hardness. In *Proceedings of the 31th Annual IEEE Symposium on Foundations of Computer Science*, pages 318–326. IEEE, 1990.

13 Ronen Gradwohl, Guy Kindler, Omer Reingold, and Amnon Ta-Shma. On the error parameter of dispersers. *Approximation, Randomization and Combinatorial Optimization: Algorithms and Techniques*, pages 294–305, 2005.

14 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.

15 Alexander D Healy. Randomness-efficient sampling within NC1. *Computational Complexity*, 17(1):3–37, 2008.

16 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bull. of the American Mathematical Society*, 43(4):439–561, 2006.

17 Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM J. Comput.*, 36(5):1231–1247, 2007.

18 A Lubotzky, R Phillips, and P Sarnak. Explicit expanders and the ramanujan conjectures. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 240–246. ACM, 1986.

19 Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

20 Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

21 Jaikumar Radhakrishnan and Amnon Ta-Shma. Tight bounds for depth-two superconcentrators. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 585–594. IEEE, 1997.

22 Anup Rao. Extractors for a constant number of polynomially small min-entropy independent sources. *SIAM Journal on Computing*, 39(1):168–194, 2009.

23 Ran Raz, Omer Reingold, and Salil P. Vadhan. Error reduction for extractors. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 191–201, 1999.

24 Omer Reingold, Ronen Shaltiel, and Avi Wigderson. Extracting randomness via repeated condensing. *SIAM Journal on Computing*, 35(5):1185–1209, 2006.

25 Aravind Srinivasan and David Zuckerman. Computing with very weak random sources. *SIAM Journal on Computing*, 28(4):1433–1459, 1999.

26 Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Loss-less condensers, unbalanced expanders, and extractors. In *Proceedings of the 33th Annual ACM Symposium on Theory of Computing*, pages 143–152. ACM, 2001.

27 Amnon Ta-Shma and David Zuckerman. Extractor codes. In *Proceedings of the 33th Annual ACM Symposium on Theory of Computing*, pages 193–199, 2001.

28 Salil Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 2011.

Pseudorandomness and Fourier Growth Bounds for Width-3 Branching Programs

Thomas Steinke*[1], Salil Vadhan†[1], and Andrew Wan‡[2]

1 **School of Engineering and Applied Sciences, Harvard University**
33 Oxford Street, Cambridge MA
{tsteinke,salil}@seas.harvard.edu

2 **Simons Institute for the Theory of Computing, UC Berkeley**
atw12@seas.harvard.edu

Abstract

We present an explicit pseudorandom generator for oblivious, read-once, width-3 branching programs, which can read their input bits in any order. The generator has seed length $\tilde{O}(\log^3 n)$. The previously best known seed length for this model is $n^{1/2+o(1)}$ due to Impagliazzo, Meka, and Zuckerman (FOCS '12). Our work generalizes a recent result of Reingold, Steinke, and Vadhan (RANDOM '13) for *permutation* branching programs. The main technical novelty underlying our generator is a new bound on the Fourier growth of width-3, oblivious, read-once branching programs. Specifically, we show that for any $f : \{0,1\}^n \to \{0,1\}$ computed by such a branching program, and $k \in [n]$,

$$\sum_{s \subseteq [n] : |s| = k} \left| \widehat{f}[s] \right| \leq n^2 \cdot (O(\log n))^k,$$

where $\widehat{f}[s] = \mathop{\mathbb{E}}_{U} \left[f[U] \cdot (-1)^{s \cdot U} \right]$ is the standard Fourier transform over $\mathbb{Z}_2^n$. The base $O(\log n)$ of the Fourier growth above is tight up to a factor of $\log \log n$.

1998 ACM Subject Classification F.1 Computation by Abstract Devices

Keywords and phrases Pseudorandomness, Branching Programs, Discrete Fourier Analysis

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.885

1 Introduction

1.1 Pseudorandom Generators for Space-Bounded Computation

A major open problem in the theory of pseudorandomness is to construct an "optimal" pseudorandom generator for space-bounded computation. That is, we want an explicit algorithm that stretches a uniformly random seed of length $O(\log n)$ to n bits that cannot be distinguished from uniform by any $O(\log n)$-space algorithm (which receives the pseudorandom bits one at a time, in a streaming fashion, and may be nonuniform). Such a generator would imply that every randomized algorithm can be derandomized with only a constant-factor increase in space (RL = L), and would also have a variety of other applications, such as in streaming algorithms [24], deterministic dimension reduction and SDP rounding [35, 15],

* Supported by NSF grant CCF-1116616 and the Lord Rutherford Memorial Research Fellowship.
† Supported in part by NSF grant CCF-1116616, US-Israel BSF grant 2010196, and a Simons Investigator Award.
‡ Part of this work was completed while at Harvard University and supported by NSF grant CCF-0964401.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 885–899

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

hashing [12], hardness amplification [21], almost k-wise independent permutations [25], and cryptographic pseudorandom generator constructions [20].

To construct a pseudorandom generator for space-bounded algorithms using space s, it suffices to construct a generator that is pseudorandom against ordered branching programs of width 2^s. A branching program[1] B is a non-uniform model of space-bounded computation that reads one input bit at a time, maintaining a state in $[w] = \{1, \ldots, w\}$, where w is called the width of B. At each time step $i = 1, \ldots, n$, B can read a different input bit $x_{\pi(i)}$ (for some permutation π) and uses a different state transition function $T_i : [w] \times \{0,1\} \to [w]$. It is often useful to think of a branching program as a directed acyclic graph consisting of $n+1$ layers of w vertices each, where the i^{th} layer corresponds to the state at time i. The transition function defines a bipartite graph between consecutive layers, where we connect state s in layer $i-1$ to states $T_i(s,0)$ and $T_i(s,1)$ in layer i (labeling those edges 0 and 1, respectively). Most previous constructions of pseudorandom generators for space-bounded computations consider *ordered* branching programs, where the input bits are read in order – that is, $\pi(i) = i$.

The classic work of Nisan [30] gave a generator with seed length $O(\log^2 n)$ that is pseudorandom against ordered branching programs of polynomial width. Despite intensive study, this is the best known seed length for ordered branching programs even of width 3, but a variety of works have shown improvements for restricted classes such as branching programs of width 2 [33, 4], and regular or permutation branching programs (of constant width) [8, 9, 26, 13, 37]. For width 3, hitting set generators (a relaxation of pseudorandom generators) have been constructed [39, 17]. The vast majority of these works are based on Nisan's original generator or its variants by Impagliazzo, Nisan, and Wigderson [23] and Nisan and Zuckerman [31], and adhere to a paradigm that seems unlikely to yield generators against general logspace computations with seed length better than $\log^{1.99} n$ (see [9]).

All known analyses of Nisan's generator and its variants rely on the order in which the output bits are fed to the branching program (given by the permutation π). The search for new ideas leads us to ask: Can we construct a pseudorandom generator whose analysis does not depend on the order in which the bits are read? A recent line of work [5, 22, 32] has constructed pseudorandom generators for unordered branching programs (where the bits are fed to the branching program in an arbitrary, fixed order); however, none match both the seed length and generality of Nisan's result. For unordered branching programs of length n and width w, Impagliazzo, Meka, and Zuckerman [22] give seed length $O((nw)^{1/2+o(1)})$ improving on the linear seed length $(1-\Omega(1)) \cdot n$ of Bogdanov, Papakonstantinou, and Wan [5].[2] Reingold, Steinke, and Vadhan [32] achieve seed length $O(w^2 \log^2 n)$ for the restricted class of *permutation* branching programs, in which $T_i(\cdot, b)$ is a permutation on $[w]$ for all $i \in [n]$ and $b \in \{0,1\}$.

Recently, a new approach for constructing pseudorandom generators has been suggested in the work of Gopalan et al. [17]; they constructed pseudorandom generators for read-once CNF formulas and combinatorial rectangles, and hitting set generators for width-3 branching programs, all having seed length $\tilde{O}(\log n)$ (even for polynomially small error). Their basic generator (e.g. for read-once CNF formulas) works by pseudorandomly partitioning the bits into several groups and assigning the bits in each group using a small-bias generator [29]. A

[1] In this work and the definition we give here, we consider read-once, oblivious branching programs, and refer to them simply as branching programs for brevity.

[2] A generator with seed length $\tilde{O}(\sqrt{n} \log w)$ is given in [32]. The generator in [22] also extends to branching programs that read their inputs more than once and in an adaptively chosen order, which is more general than the model we consider.

key insight in their analysis is that the small-bias generator only needs to fool the function "on average," where the average is taken over the possible assignments to subsequent groups, which is a weaker requirement than fooling the original function or even a random restriction of the original function. (For a more precise explanation, see Section 4.)

The analysis of Gopalan et al. [17] does not rely on the order in which the output bits are read, and the previously mentioned work by Reingold, Steinke, and Vadhan [32] uses Fourier analysis of branching programs to show that the generator of Gopalan et al. fools unordered permutation branching programs. In this work we further develop Fourier analysis of branching programs and show that the pseudorandom generator of Gopalan et al. with seed length $\tilde{O}(\log^3 n)$ fools width-3 branching programs:

▶ **Theorem 1** (Main Result). *There is an explicit pseudorandom generator* $G : \{0,1\}^{O(\log^3 n \cdot \log\log n)} \to \{0,1\}^n$ *fooling oblivious, read-once (but unordered), branching programs of width* 3 *and length* n.

The previous best seed length for this model is the aforementioned length of $O(n^{1/2+o(1)})$ given in [22]. The construction of the generator in Theorem 1 is essentially the same as the generator of Gopalan et al. [17] for read-once CNF formulas, which was used by Reingold et al. [32] for permutation branching programs. In our analysis, we give a new bound on the Fourier mass of width-3 branching programs.

1.2 Fourier Growth of Branching Programs

For a function $f : \{0,1\}^n \to \mathbb{R}$, let $\widehat{f}[s] = \mathop{\mathbb{E}}_{U}\left[f[U] \cdot (-1)^{s \cdot U}\right]$ be the standard Fourier transform over $\mathbb{Z}_2^n$, where U is a random variable distributed uniformly over $\{0,1\}^n$ and $s \subseteq [n]$ or, equivalently, $s \in \{0,1\}^n$. The Fourier mass of f (also called the spectral norm of f), defined as $L(f) := \sum_{s \neq \emptyset} |\hat{f}[s]|$, is a fundamental measure of complexity for Boolean functions (e.g., see [18]), and its study has applications to learning theory [27, 28], communication complexity [19, 1, 38, 34], and circuit complexity [7, 10, 11]. In the study of pseudorandomness, it is well-known that small-bias generators[3] with bias ε/L (which can be sampled using a seed of length $O(\log(n \cdot L/\varepsilon))$ [29, 2]) will ε-fool any function whose Fourier mass is at most L. Width-2 branching programs have Fourier mass at most $O(n)$ [4, 33] and are thus fooled by small-bias generators with bias ε/n. Unfortunately, such a bound does not hold even for very simple width-3 programs. For example, the 'mod 3 function,' which indicates when the hamming weight of its input is a multiple of 3 has Fourier mass exponential in n.

However, a more refined measure of Fourier mass is possible and often useful: Let $L^k(f) = \sum_{|s|=k} |\hat{f}[s]|$ be the level-k Fourier mass of f. A bound on the Fourier growth of f, or the rate at which $L^k(f)$ grows with k, was used by Mansour [28] to obtain an improved query algorithm for polynomial-size DNF; the junta approximation results of Friedgut [16] and Bourgain [6] are proven using approximating functions that have slow Fourier growth. This notion turns out to be useful in the analysis of pseudorandom generators as well: Reingold et al. [32] show that the generator of Gopalan et al. [17] will work if there is a good bound on the Fourier mass of low-order coefficients. More precisely, they show that for any class $\mathcal{C}$ of functions computed by branching programs that is closed under restrictions and decompositions and satisfies $L^k(f) \leq \mathrm{poly}(n) \cdot c^k$ for every k and $f \in \mathcal{C}$, there is a

[3] A small-bias generator with bias μ outputs a random variable $X \in \{0,1\}^n$ such that $\left|\mathop{\mathbb{E}}_{X}\left[(-1)^{s \cdot X}\right]\right| \leq \mu$ for every $s \subset [n]$ with $s \neq \emptyset$.

pseudorandom generator with seed length $\tilde{O}(c \cdot \log^2 n)$ that fools every $f \in \mathcal{C}$. They then bound the Fourier growth of permutation branching programs (and the even more general model of "regular" branching programs, where each layer is a regular bipartite graph) to obtain a pseudorandom generator for permutation branching programs:

▶ **Theorem 2** ([32, Theorem 1.4]). *Let $f : \{0,1\}^n \to \{0,1\}$ be computed by a length-n, width-w, read-once, oblivious,* regular *branching program. Then, for all $k \in [n]$, $L^k(f) \leq (2w^2)^k$.*

In particular, the mod 3 function over $O(k)$ bits, which is computed by a permutation branching program of width 3, has Fourier mass $2^{\Theta(k)}$ a level k. However, the Tribes function,[4] which is also computed by a width-3 branching program, has Fourier mass $\Omega(\log n / \log k)^k$ at level k, so the bound in Theorem 2 does not hold for non-regular branching programs even of width 3.

The Coin Theorem of Brody and Verbin [9] implies a related result: essentially, a function computed by a width-w, length-n branching program cannot distinguish product distributions on $\{0,1\}^n$ any better than a function satisfying $L^k(f) \leq (\log n)^{O(wk)}$ for all k. To be more precise, if $X \in \{0,1\}^n$ is n independent samples from a coin with bias β (that is, each bit has expectation $(1+\beta)/2$), then $\mathop{\mathbb{E}}_X [f[X]] = \sum_s \widehat{f}[s]\beta^{|s|}$. If $L^k(f) \leq (\log n)^{O(wk)}$ for all k, then

$$\left| \mathop{\mathbb{E}}_X [f[X]] - \mathop{\mathbb{E}}_U [f[U]] \right| = \left| \sum_{s \neq 0} \widehat{f}[s]\beta^{|s|} \right| \leq \sum_{k \in [n]} L^k(f)|\beta|^{|s|} \leq O(|\beta|(\log n)^{O(w)}),$$

assuming $|\beta| \leq 1/(\log n)^{O(w)}$. Brody and Verbin prove that, if f is computed by a length-n, width-w branching program, then $|\mathop{\mathbb{E}}_X [f[X]] - \mathop{\mathbb{E}}_U [f[U]]| \leq O(|\beta|(\log n)^{O(w)})$. Since distinguishing product distributions captures much of the power of a class of functions, this leads to the following conjecture.

▶ **Conjecture 3** ([32, Conjecture 8.1]). *For every constant w, the following holds. Let $f : \{0,1\}^n \to \{0,1\}$ be computed by a width-w, read-once, oblivious branching program. Then*

$$L^k(f) \leq n^{O(1)} \cdot (\log n)^{O(k)},$$

where the constants in the $O(\cdot)$s may depend on w.

In this work, we prove this conjecture for $w = 3$:

▶ **Theorem 4** (Fourier Growth of Width 3). *Let $f : \{0,1\}^n \to \{0,1\}$ be computed by a width-3, read-once, oblivious branching program. Then, for all $k \in [n]$,*

$$L^k(f) := \sum_{s : |s| = k} |\widehat{f}[s]| \leq n^2 \cdot (O(\log n))^k .$$

This bound is the main contribution of our work and, when combined with the techniques of Reingold et al. [32], implies our main result (Theorem 1).

The Tribes function of [3] shows that the base of $O(\log n)$ of the Fourier growth in Theorem 4 is tight up to a factor of $\log \log n$. (See the full version of this paper for a proof.)

We also prove Conjecture 3 with $k = 1$ for any constant width w:

[4] The Tribes function (introduced by Ben-Or and Linial [3]) is a DNF formula where all the terms are the same size and every input appears exactly once. The size of the clauses in this case is chosen to give an asymptotically constant acceptance probability on uniform input.

▶ **Theorem 5.** *Let $f : \{0,1\}^n \to \{0,1\}$ be computed by a width-w, length-n, read-once, oblivious branching program. Then*

$$L^1(f) = \sum_{i \in [n]} |\widehat{f}[\{i\}] \leq (O(\log n))^{w-2}.$$

1.3 Techniques

The intuition behind our approach begins with two extreme cases of width-3 branching programs: permutation branching programs and branching programs in which every layer is a non-permutation layer. Permutation branching programs "mix" well: on a uniform random input, the distribution over states gets closer to uniform (in ℓ_2 distance) in each layer. We can use this fact with an inductive argument to achieve a bound of $2^{O(k)}$ on the level-k Fourier mass (this is the bound of Theorem 2).

For branching programs in which *every* layer is a non-permution layer, we can make use of an argument from the work of Brody and Verbin [9]: when we apply a random restriction (where each variable is kept free with probability roughly $1/k$) to such a branching program, the resulting program is 'simple' in that the width has collapsed to 2 in many of the remaining layers. This allows us to use arguments tailored to width-2 branching programs, which are well-understood. In particular, we can use the same concept of mixing as used for permutation branching programs.

To handle general width-3 branching programs, which may contain an arbitrary mix of permutation and non-permutation layers, we group the layers into "chunks" containing exactly one non-permutation layer each. Instead of using an ordinary random restriction, we consider a series of restrictions similar to those in Steinberger's "interwoven hybrids" technique [36] (in our argument each chunk will correspond to a single layer in [36]). In Section 3.1, we use such restrictions to show that the level-k Fourier mass of an arbitrary width-3 program can be bounded in terms of the level-k Fourier mass of a program D which has the following "pseudomixing" form: D can be split into $r \in [n]$ branching programs $D_1 \circ D_2 \circ \cdots \circ D_r$, where each D_i has at most $3k$ non-regular layers and the layer splitting consecutive D_is has width 2.

We then generalize the arguments used for width-2 branching programs to "pseudomixing" branching programs. We can show that each chunk D_i is either mixing or has small Fourier growth, which suffices to bound the Fourier growth of D.

2 Preliminaries

2.1 Branching Programs

We view a length-n, width-w **branching program** as a function $B : \{0,1\}^n \times [w] \to [w]$, which takes a start state $u \in [w]$ and an input string $x \in \{0,1\}^n$ and outputs a final state $B[x](u)$. We can view B as computing a Boolean function by fixing a start state and set of accept states in $[w]$. In this work we consider branching programs with random (or pseudorandom) inputs, in which case a program can be viewed as a Markov chain randomly taking initial states to final states. That is, B can be viewed as a matrix-valued function $B : \{0,1\}^n \to \{0,1\}^{w \times w}$ where $B[x]_{u,v} = 1$ if and only if $B[x](u) = v$. For a random variable X on $\{0,1\}^n$, we have $\mathop{\mathbb{E}}_X [B[X]] \in [0,1]^{w \times w}$, where the entry in the u^{th} row and v^{th} column $\mathop{\mathbb{E}}_X [B[X]]_{u,v}$ is the probability that B takes the initial state u to the final state v when given a random input from the distribution X. The matrix $\mathop{\mathbb{E}}_X [B[X]]$ is **stochastic**, that is, its

rows give probability distributions (i.e., they are non-negative and sum to 1). A **regular program** B has the property that the uniform distribution is a stationary distribution of the Markov chain $\mathop{\mathbb{E}}_{U}[B[U]]$, whereas, if B is a **permutation program**, the uniform distribution is stationary for $\mathop{\mathbb{E}}_{X}[B[X]]$ for *any* distribution X.

We write $B_1 \circ B_2$ to denote the *concatenation* of two branching programs, where the start state of B_2 is determined by the final state of B_1 on the input. Thus the matrix representation of $B_1 \circ B_2[x_1 \circ x_2]$ is given by $B_1[x_1] \cdot B_2[x_2]$. A length-n, width-w, **ordered branching program** (abbreviated **OBP**) is a program B that can be written $B = B_1 \circ B_2 \circ \cdots \circ B_n$, where each B_i is a length-1 width-w program. We refer to B_i as the i^{th} **layer** of B. We denote the **subprogram** of B from layer i to layer j by $B_{i\cdots j} := B_i \circ B_{i+1} \circ \cdots \circ B_j$. We sometimes consider branching programs of varying width – some layers have more vertices than others. The overall width of the program is the maximum width of any layer. This means that the edge layers B_i may give non-square matrices. For $i \in [n]$, if $B_i[x] \in \{0,1\}^{w \times w'}$, then we refer to w as the width of layer $i-1$ and w' as the width of layer i.

2.2 Fourier Analysis

Let $B : \{0,1\}^n \to \mathbb{R}^{w \times w'}$ be a matrix-valued function (such as given by a length-n, width-w branching program). Then we define the **Fourier transform** of B as a matrix-valued function $\widehat{B} : \{0,1\}^n \to \mathbb{R}^{w \times w'}$ given by

$$\widehat{B}[s] := \mathop{\mathbb{E}}_{U}\left[B[U]\chi_s(U)\right],$$

where $s \in \{0,1\}^n$ (or, equivalently, $s \subset [n]$) and

$$\chi_s(x) = (-1)^{\sum_i x_i \cdot s_i} = \prod_{i \in s}(-1)^{x(i)}.$$

We refer to $\widehat{B}[s]$ as the s^{th} **Fourier coefficient** of B, which has **order** (or degree) $|s|$. Note that this is equivalent to taking the real-valued Fourier transform of each of the $w \cdot w'$ entries of $B[x]$ separately, but we see in the following lemma that this matrix-valued Fourier transform is nicely compatible with matrix algebra.

▶ **Lemma 6.** *Let $A : \{0,1\}^n \to \mathbb{R}^{w \times w'}$ and $B : \{0,1\}^n \to \mathbb{R}^{w' \times w''}$ be matrix valued functions. Let X, Y, and U be independent random variables over $\{0,1\}^n$, where U is uniform. Let $s, t \in \{0,1\}^n$. Then we have the following.*

- *Decomposition: If $C[x \circ y] = A[x] \cdot B[y]$ for all $x, y \in \{0,1\}^n$, then $\widehat{C}[s \circ t] = \widehat{A}[s] \cdot \widehat{B}[t]$.*
- *Fourier Inversion for Matrices: $B[x] = \sum_s \widehat{B}[s]\chi_s(x)$.*
- *Parseval's Identity: $\sum_{s \in \{0,1\}^n} \left|\left|\widehat{B}[s]\right|\right|_{Fr}^2 = \mathop{\mathbb{E}}_{U}\left[||B[U]||_{Fr}^2\right]$.*

The Decomposition property is what makes the matrix-valued Fourier transform more convenient than separately taking the Fourier transform of the matrix entries as done by Bogdanov et al. [5]. If B is a length-n, width-w, ordered branching program, then, for all $s \in \{0,1\}^n$,

$$\widehat{B}[s] = \widehat{B}_1[s_1] \cdot \widehat{B}_2[s_2] \cdot \cdots \cdot \widehat{B}_n[s_n].$$

The **Fourier mass** of a matrix-valued function B is $L(B) := \sum_{s \neq 0} \left|\left|\widehat{B}[s]\right|\right|_2$, and the Fourier mass at level-k is $L^k(B) := \sum_{s \in \{0,1\}^n : |s| = k} \left|\left|\widehat{B}[s]\right|\right|_2$. We define $L^{\geq k}(B) := \sum_{k' \geq k} L^{k'}(B)$ and $L^{\leq k}(B)$, $L^{>k}(B)$, $L^{<k}(B)$ are defined analogously. The Fourier mass is unaffected by order:

▶ **Lemma 7.** *Let $B, B' : \{0,1\}^n \to \mathbb{R}^{w \times w}$ be matrix-valued functions satisfying $B[x] = B'[\pi(x)]$, where $\pi : [n] \to [n]$ is a permutation. Then, for all $s \in \{0,1\}^n$, $\widehat{B}[s] = \widehat{B'}[\pi(s)]$. In particular, $L(B) = L(B')$ and $L^k(B) = L^k(B')$ for all k.*

Lemma 7 implies that the Fourier mass of any read-once, oblivious branching program is equal to the Fourier mass of the corresponding ordered branching program.

3 Fourier Analysis of Width-3 Branching Programs

In this section we sketch the proof of our bound on the low-order Fourier mass of width-3, read-once, oblivious branching programs (Theorem 4). This is key to the analysis of our pseudorandom generator. See the full version of our paper (which is available online) for a complete proof.

To prove Theorem 4 we will consider the matrix valued function B of the branching program computing f. Note that $|\hat{f}[s]| \leq ||\widehat{B}[s]||_2$ for all s so $L^k(f) \leq L^k(B)$. We may also assume without loss of generality that the first and last layers of the program have width 2 (there is only one start state, and there are at most 2 accept states otherwise the program is trivial). The proof proceeds in two parts. The first part reduces the problem to one about branching programs of a special form, namely ones where many layers have been reduced to width-2. The second part uses the mixing properties of width-2 programs to bound the Fourier mass.

3.1 Part 1 – Reduction of Width by Random Restriction

First some definitions:

For $g \subset [n]$ and $x \in \{0,1\}^n$, define the **restriction of B to g using x – denoted** $B|_{\overline{g} \leftarrow x}$ – to be the branching program obtained by setting the inputs (layers of edges) of B outside g to values from x and leaving the inputs in g free. More formally,

$$B|_{\overline{g} \leftarrow x}[y] = B[\mathrm{Select}(g, y, x)], \quad \text{where} \quad \mathrm{Select}(g, y, x)_i = \left\{ \begin{array}{ll} y_i & i \in g \\ x_i & i \notin g \end{array} \right\}.$$

Our reduction can be stated as follows.

▶ **Proposition 8.** *Let B be a length-n width-3 ordered branching program (abbreviated **3OBP**), $m \geq k$, and $k \in [n]$ with the first and last layers having width at most 2. Then*

$$L^k(B) \leq n \cdot \binom{m}{k} \sum_{\ell \geq 0} 2^{-\ell(m-k)} L^k(D^{6(\ell+1)k})$$

where each $D^{6(\ell+1)k} = D_1^{6(\ell+1)k} \circ D_2^{6(\ell+1)k} \circ \cdots \circ D_r^{6(\ell+1)k}$, where $r \in [n]$, each $D_i^{6(\ell+1)k}$ is a 3OBP with at most $6(\ell+1)k$ non-regular layers, and the first and last layers of each D_i have width at most 2.

In Section 3.2, we will prove $L^k(D^{6(\ell+1)k}) \leq n \cdot O(\ell)^k$. Taking $m = 2k$, this implies $L^k(B) \leq n^2 \cdot O(k)^k$. Finally, we show that we may assume $k \leq O(\log n)$, so we get a Fourier growth bound of $L^k(B) \leq n^2 \cdot O(\log n)^k$. Here we focus on the proof of Proposition 8.

Define a **chunk** to be a 3OBP with exactly one non-regular layer. An ***l*-chunk** 3OBP is a 3OBP B such that $B = C_1 \circ C_2 \circ \cdots \circ C_l$, where each C_i is a chunk. Equivalently, an l-chunk 3OBP is a 3OBP with exactly l non-regular layers. The partitioning of B into chunks is not necessarily unique. But we fix one such partitioning for each 3OBP and simply

refer to the i^{th} chunk C_i. If B is an l-chunk length-n 3OBP, let $c_i \subset [n]$ be the coordinates corresponding to C_i.

We will compute a bound on the level-k Fourier weight of B via a series of "interwoven" restrictions similar to Steinberger's technique [36]. Lemma 9 below tells us that we may obtain a bound by bounding, in expectation, the level-k weight of a restricted branching program. We then argue that with high probability over this restriction, the width of the resulting program will be essentially reduced. In particular, there is a layer of width 2 after every $O(m)$ non-regular layers.

We now describe some notation that will be used for the interwoven restrictions. For $t \subset [m]$, define

$$g_t := \bigcup_{(i \bmod m)+1 \in t} c_i \quad \text{and} \quad G_t^k := \{s \subset g_t : |s| = k\}.$$

We refer to g_t as the **t^{th} group of indices** and G_t^k as the **t^{th} group of (order k) Fourier coefficients**. The following simple lemma tells us that we may bound the level-k Fourier weight by considering a fixed subset $t \subset [m]$ of size k and the level-k Fourier weight of the branching program that results by randomly restricting the variables outside of g_t:

▶ **Lemma 9.** *Let B be a length-n 3OBP, $k \in [n]$, $m \geq k$ and g_t as above. Then*

$$L^k(B) \leq \binom{m}{k} \max_{t \subset [m]:|t|=k} \mathop{\mathbb{E}}_{U} \left[L^k(B|_{\overline{g_t} \leftarrow U})\right].$$

Given Lemma 9, we may now prove Proposition 8 by giving an upper bound on $\mathop{\mathbb{E}}_{U} \left[L^k(B|_{\overline{g_t} \leftarrow U})\right]$ for a fixed $t \subset [m]$ with $|t| = k$. To do this, we prove that a random restriction to $\overline{g_t}$ will, with high probability, result in a branching program of the desired form.

▶ **Lemma 10.** *Let B be a length-n 3OBP, $k, \ell \in [n]$, $m \geq k$ and fix $t \subseteq [m]$ with $|t| = k$. Then with probability at least $1 - n \cdot 2^{-\ell(m-k)}$ over a random choice of $x \in \{0,1\}^n$,*

$$B|_{\overline{g_t} \leftarrow x} = D_1 \circ D_2 \circ \cdots \circ D_r,$$

where $r \in [n]$ and each D_i is a 3OBP with at most $6\ell k$ non-regular layers and the layer of vertices between D_{i-1} and D_i have width at most 2.

3.2 Part 2 – Mixing in Width-2

Now it remains to bound the Fourier mass of 3OBPs of the form given by Proposition 8.

▶ **Proposition 11.** *Let D^ℓ be a length-n 3OBP such that*

$$D^\ell = D_1^\ell \circ D_2^\ell \circ \cdots \circ D_r^\ell,$$

where each D_i^ℓ is a 3OBP with at most ℓ non-regular layers and width 2 in the first and last layers. Then $L^k(D^\ell) \leq 2n \cdot (6000(\ell+1))^k$ for all k.

A key notion in our proof is a measure of the extent to which a branching program (or subprogram) mixes, and the way this is reflected in the Fourier spectrum. For an ordered branching program D of width w, define

$$\lambda(D) = \max_{x \in \mathbb{R}^w : \sum_i x_i = 0} \frac{\left|\left| x \mathop{\mathbb{E}}_{U} [D[U]] \right|\right|_2}{||x||_2}.$$

The quantity $\lambda(D)$ is a measure of the **mixing** of D. If D is regular, we have $\lambda(D) \in [0,1]$, where 0 corresponds to perfect mixing and 1 to no mixing. If D is not regular, it is possible that $\lambda(D) > 1$. However, for width-2 – where $\mathop{\mathbb{E}}_{U}[D[U]]$ is a 2×2 matrix – it turns out that $\lambda(D) \leq 1$ even if D is non-regular. In particular,

$$\text{if } \mathop{\mathbb{E}}_{U}[D[U]] = \begin{pmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{pmatrix}, \quad \text{then } \lambda(D) = \frac{\left|\left|(1,-1)\mathop{\mathbb{E}}_{U}[D[U]]\right|\right|_2}{||(1,-1)||_2} = |1-\alpha-\beta|.$$

The rows of $\mathop{\mathbb{E}}_{U}[D[U]]$ must sum to 1 and have non-negative entries (as they are a probability distribution). So $\alpha, \beta \in [0,1]$, which implies $\lambda(D) \leq 1$. This fact is crucial to our analysis and is the main reason our results do not extend to higher widths.

Note that for any $s \neq 0$, the rows of $\widehat{D}[s]$ sum to zero. Thus for any branching program $D = D_1 \circ D_2$ and coefficient $\widehat{D}[s]$ with $s = (s_1, s_2)$ satisfying $s_2 = 0$, we have

$$\left|\left|\widehat{D}[s]\right|\right|_2 \leq \left|\left|\widehat{D_1}[s_1]\right|\right|_2 \cdot \lambda(D_2).$$

For branching programs B in which every layer is mixing – that is $\lambda(B_i) \leq C < 1$ for all i – this fact can be used with an inductive argument (simpler than the proof below) to obtain a $O(n)/(1-C)^{O(k)}$ bound on the level-k Fourier mass. We show that any D_i in the branching program of the form given by Proposition 8 will either mix well or have small Fourier mass after restriction. More precisely, define the **p-damped Fourier mass** of a branching program B as

$$L_p(B) = \sum_{k>0} p^k L^k(B) = \sum_{s \neq 0} p^{|s|} \left|\left|\widehat{B}[s]\right|\right|_2.$$

Note that $L^k(B) \leq L_p(B) p^{-k}$ for all k and p. The main lemma we prove in this section is the following.

▶ **Lemma 12.** *If D is a length-d 3OBP with $k \geq 1$ non-regular layers that has only two vertices in the first and last layers, then*

$$\lambda(D) + L_p(D) \leq 1$$

for any $p \leq 1/6000(k+1)$.

First, we show that Lemma 12 implies Proposition 11:

Proof of Proposition 11. We inductively show that

$$L_p(D_1^\ell \circ \cdots \circ D_i^\ell) \leq 2i,$$

and hence $L_p(D) \leq 2r \leq 2n$. For $i = 0$ this is trivial. Now suppose it holds for i. By decomposition (Lemma 6), we have

$$\begin{aligned} L_p(D_1^\ell \cdots D_i^\ell \circ D_{i+1}^\ell) \leq & L_p(D_1^\ell \cdots D_i^\ell) \cdot L_p(D_{i+1}^\ell) + L_p(D_1^\ell \cdots D_i^\ell)\lambda(D_{i+1}^\ell) \\ & + \left|\left|\widehat{D_1^\ell \cdots D_i^\ell}[0]\right|\right|_2 \cdot L_p(D_{i+1}^\ell) \\ \leq & L_p(D_1^\ell \cdots D_i^\ell) \cdot 1 + \sqrt{2} L_p(D_{i+1}^\ell) \leq 2i + 2. \end{aligned}$$

The first inequality follows from mixing and the second from Lemma 12. Thus, we have that $L^k(D^\ell) \leq p^{-k} L_p(D^\ell) \leq 2n \cdot (6000(\ell+1))^k$, as required ◀

Now we turn our attention to Lemma 12. We split into two cases: If $\lambda(D)$ is far from 1 i.e. $\lambda(D) \leq 0.99$, then we need only ensure $L_p(D) \leq 1/100$. This is the 'easy case' which proceeds much like the analysis of regular branching programs [32]. If $\lambda(D) = 1$, then D is trivial – i.e. $L_p(D) = 0$ – and we are also done. The 'hard case' is when $\lambda(D)$ is very close to 1. i.e. $0.99 \leq \lambda(D) < 1$.

Easy Case – Good Mixing

The argument used by Reingold et al. [32] for regular branching program can be extended to give the following.

▶ **Lemma 13.** *Let D be a 3OBP with at most k non-regular layers. If $p \leq 1/6000(k+1)$, then $L_p(D) \leq 1/100$.*

It immediately follows that $\lambda(D)+L_p(D) \leq 1$ when $p \leq 1/6000(k+1)$, assuming $\lambda(D) < 0.99$. This covers the 'easy' case of Lemma 12.

Hard Case – Poor Mixing

Now we consider the case where $\lambda(D) \in [0.99, 1]$.

▶ **Lemma 14.** *Let D be a 3OBP with at most k non-regular layers where the first and last layers of vertices have width 2. Suppose $\lambda(D) \in [0.99, 1]$. If $p \leq 1/(24k+12)$, then $L_p(D) + \lambda(D) \leq 1$.*

This covers the 'hard' case of Lemma 12 and, along with Lemma 13 completes the proof of Lemma 12.

Since D has width 2 in the first and last layers, we view $D[x]$ as a 2×2 matrix. We can write the expectation (which is stochastic) as

$$\mathop{\mathbb{E}}_{U}[D[U]] = \begin{pmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{pmatrix}.$$

We can assume (by permuting rows and columns) that $\lambda(D) = 1-\alpha-\beta$ and $\alpha, \beta \in [0, 1/100]$. Now write

$$D[x] = \begin{pmatrix} 1-f(x) & f(x) \\ g(x) & 1-g(x) \end{pmatrix},$$

where $f, g : \{0,1\}^d \to \{0,1\}$. Then $\alpha = \mathop{\mathbb{E}}_{U}[f(U)]$ and $\beta = \mathop{\mathbb{E}}_{U}[g(U)]$. We can view D has having two *corresponding* start and end states. The probability that, starting in the first start state, we end in the first end state is $1-\alpha \geq 0.99$. Likewise, the probability that, starting in the second start state, we end in the second end state is $1-\beta \geq 0.99$. The function f is computed by starting in the first start state and accepting if we end in the second end state – that is, if we "cross over". Likewise, g computes the function telling us whether we will cross over from the second start state to the first end state. Intuitively, there is a low (1/100) probability of crossing over, so the program behaves like two disjoint programs.

We will show that $L_p(f) \leq (12k+6)p\alpha$ and $L_p(g) \leq (12k+6)p\beta$ for $p \leq 1/(6k+3)$, from which the result follows by choosing p such that $L_p(f) \leq \alpha/2$ and $L_p(g) \leq \beta/2$.

The plan is as follows.

1. Show that we can partition the vertices of D into two sets with $O(k)$ edges crossing between the sets such that each layer has at least one vertex in each set. Intuitively, this partitions D into two width-2 branching programs with a few edges going between them.

2. Using this partition, show that we can write $f(x) = \sum_s \prod_j f_{s,j}(x_j)$, where each $f_{s,j}$ is a $\{0,1\}$-valued function computed by a *regular* width-2 branching program, the product is over $O(k)$ terms and the x_js are a partition of x.
3. Let $f_s(x) = \prod_j f_{s,j}(x_j)$ and $\alpha_s = \mathop{\mathbb{E}}_U[f_s(U)]$. Show that $L_p(f_s) \leq (12k+6)p\alpha_s$ for $p \leq 1/(6k+3)$. Then

$$L_p(f) \leq \sum_s L_p(f_s) \leq \sum_s (12k+6)p\alpha_s \leq (12k+6)p\alpha.$$

The same holds for g, which gives the result. Formally, these steps proceed as follows. See the full version of this paper for proofs of these lemmas.

Step 1

▶ **Lemma 15.** *Let D be a 3OBP with at most k non-regular layers and width-2 in the first and last layers of vertices. Suppose $\lambda(D) \in [0.99, 1]$. Then there is a partition of the vertices of D such that each layer has at least one vertex in each side of the partition and there are at most $2k+1$ layers with an edge that crosses the partition.*

There are two possible start states in the first layer. We partition the vertices based on whether they are more likely to be reached if we start at the first start state versus starting from the second start state. The $\lambda(D) \geq 0.99$ assumption tells us that this partition is very strong, in the sense that most vertices are much more likely to be visited from one start state than from the other. Consequently there are few edges crossing the partition.

Step 2

▶ **Lemma 16.** *Let D be a length-d 3OBP with at most k non-regular layers and width-2 in the first and last layers of vertices. Suppose $\lambda(D) \geq 0.99$. If $f : \{0,1\}^n \to \{0,1\}$ is the function computed by D, then we can write $f(x) = \sum_s \prod_j f_{s,j}(x_j)$, where each $f_{s,j}$ is computed by a regular width-2 ordered branching program and the x_j's are a partition of x into at most $6k+3$ parts.*

To prove this, we use the partition of Lemma 15. Intuitively, D is partitioned into two width-2 branching programs. The problem is the $O(k)$ layers where D is either non-regular or there is an edge crossing the partition – call these critical layers. We condition on what happens at the critical layers, which we can express with a width-2 program, and finally, we express f by summing over all possibilities for what happens in the critical layers. The product appears because we must use an AND that the event we are conditioning on is true.

Step 3

Now we have reduced the problem to analysing functions of a very simple form. We can use the basic properties of width-2 branching programs to prove the following, which suffices to prove Lemma 14.

▶ **Lemma 17.** *Let $f : \{0,1\}^n \to \{0,1\}$ be of the form $f(x) = \prod_{j \in [k]} f_j(x_j)$, where the x_js are a partition of x and each f_j is computed by a width-2 ordered regular branching program. Then $L_p(f) \leq 2kp \cdot \mathop{\mathbb{E}}_U[f(U)]$ for any $p \leq 1/k$.*

4 The Pseudorandom Generator

Our main result Theorem 1 follows from plugging our Fourier growth bound (Theorem 4) into the analysis of [32]. We include a general statement here for completeness:

▶ **Theorem 18.** *Let $\mathcal{C}$ be a set of ordered branching programs of length at most n and width at most w that is closed under restrictions and subprograms – that is, if $B \in \mathcal{C}$, then $B|_{t \leftarrow x} \in \mathcal{C}$ for all t and x and $B_{i \cdots j} \in \mathcal{C}$ for all i and j. Suppose that, for all $B \in \mathcal{C}$ and $k \in [n]$, we have $L^k(B) \leq ab^k$, where $b \geq 2$. Let $\varepsilon > 0$.*

Then there exists a pseudorandom generator $G_{a,b,n,\varepsilon} : \{0,1\}^{s_{a,b,n,\varepsilon}} \to \{0,1\}^n$ with seed length $s_{a,b,n,\varepsilon} = O\left(b \cdot \log(b) \cdot \log(n) \cdot \log\left(\frac{abwn}{\varepsilon}\right)\right)$ such that, for any length-n, width-w, read-once, oblivious (but unordered) branching program B that corresponds to an ordered branching program in $\mathcal{C}$,[5]

$$\left\| \mathop{\mathbb{E}}_{U_{s_{a,b,n,\varepsilon}}} \left[B[G_{a,b,n,\varepsilon}(U_{s_{a,b,n,\varepsilon}})]\right] - \mathop{\mathbb{E}}_{U}\left[B[U]\right] \right\|_2 \leq \varepsilon.$$

Moreover, $G_{a,b,n,\varepsilon}$ can be computed in space $O(s_{a,b,n,\varepsilon})$.

To prove Theorem 1 we set $\mathcal{C}$ to be the class of all 3OBPs of length at most n. Theorem 4 gives a bound corresponding to $a = \text{poly}(n)$ and $b = O(\log n)$. This gives the required generator. The statements of Theorems 1 and 18 differ in that Theorem 18 bounds the error of the pseudorandom generator with respect to a matrix-valued function, while Theorem 1 bounds the error with respect to a $\{0,1\}$-valued function. These statements are equivalent as the $\{0,1\}$-valued function is simply one entry in the matrix-valued function.

The pseudorandom generator is formally defined as follows.

Algorithm for $G_{a,b,n,\varepsilon} : \{0,1\}^{s_{a,b,n,\varepsilon}} \to \{0,1\}^n$.

Parameters: $n \in \mathbb{N}$, $\varepsilon > 0$.

Input: A random seed of length $s_{a,b,n,\varepsilon}$.

1. Compute appropriate values of $p \leq 1/2b$, $\varepsilon' = \varepsilon p/14w\log_2(n)$, $k \geq \log_2\left(8an^4w/\varepsilon'\right)$, $\delta \leq \varepsilon'(p/2)^{2k}$, and $\mu \leq \varepsilon'/2ab^k$. [6]
2. If $n \leq 320 \cdot \lceil \log_2(1/\varepsilon') \rceil / p$, output n truly random bits and stop.
3. Sample $T \in \{0,1\}^n$ where each bit has expectation p and the bits are δ-almost $2k$-wise independent.
4. If $|T| < pn/2$, output 0^n and stop.
5. Recursively sample $\tilde{U} \in \{0,1\}^{\lfloor n(1-p/2) \rfloor}$. i.e. $\tilde{U} = G_{a,b,\lfloor n(1-p/2) \rfloor,\varepsilon}(U)$.
6. Sample $X \in \{0,1\}^n$ from a μ-biased distribution.
7. Output $\text{Select}(T, X, \tilde{U}) \in \{0,1\}^n$.[7]

The analysis of this generator can be found in the full version of this paper.

---

[5] That is, there exists $B' \in \mathcal{C}$ and a permutation of the bits $\pi : \{0,1\}^n \to \{0,1\}^n$ such that $B[x] = B'[\pi(x)]$ for all x.

[6] For the purposes of the analysis we assume that ε', k, p, δ, and μ are the same at every level of recursion. So if $G_{a,b,n,w,\varepsilon}$ is being called recursively, use the same values of ε', p, k, δ, and μ as at the previous level of recursion. We pick values within a constant factor of these constraints.

[7] Technically, we must pad $\tilde{U}$ with zeros in the locations specified by T (i.e. $\tilde{U}_i = 0$ for $i \in T$) to obtain the right length.

5 Further Work

Our results hinge on the fact that "mixing" is well-understood for regular branching programs [8, 32, 26, 13, 37] and for (non-regular) width-2 branching programs [4]. We are able to use random restrictions to reduce from width 3 to width 2 (Section 3.1), where we can exploit our understanding of mixing (Section 3.2). Indeed, this understanding underpins most results for these restricted models of branching programs.

What about width 4 and beyond? Using a random restriction we can reduce analysing width 4 to "almost" width 3 – that is, Proposition 8 generalises. Unfortunately, the reduction does not give a true width-3 branching program and thus we cannot repeat the reduction to width 2. Moreover, we have a poor understanding of mixing for non-regular width-3 branching programs, which means we cannot use the same techniques that have worked for width-2 branching programs.

Our results provide some understanding of mixing in width-3. We hope this understanding can be developed further and will lead to proving Conjecture 3 and other results.

The biggest obstacle to extending our techniques to $w > 3$ is Lemma 12. The problem is that the parameter $\lambda(D)$ is no longer a useful measure of mixing for width-3 and above. In particular, $\lambda(D) > 1$ is possible if $\mathop{\mathbb{E}}_{U}[D[U]]$ is a 3×3 matrix. To extend our techniques, we need a better notion of mixing. Using $\lambda(D)$ is useful for regular branching programs (it equals the second eigenvalue for regular programs), but is of limited use for non-regular branching programs. Our proof uses a different notion of mixing – collisions: To prove Proposition 8, we used the fact that a random restriction of a non-regular layer will with probability at least $1/2$ result in the width of the right side of the layer being reduced. This is a form of mixing, but it is not captured by λ. Ideally, we want a notion of mixing that captures both λ and width-reduction under restrictions.

Our proofs combine the techniques of Braverman et al. [8] and those of Brody and Verbin [9] and Steinberger [36]. We would like to combine them more cleanly – presently the proof is split into two parts (Proposition 8 and Lemma 12). This would likely involve developing a deeper understanding of the notion of mixing.

Our seed length $\tilde{O}(\log^3 n)$ is far from the optimal $O(\log n)$. Further improvement would require some new techniques:

We could potentially relax our notion of Fourier growth to achieve better results. Rather than bounding $L^k(f)$, it suffices to bound $L^k(g)$, where g *approximates* f:

▶ **Proposition 19** ([14, Proposition 2.6]). *Let $f, f_+, f_- : \{0,1\}^n \to \mathbb{R}$ satisfy $f_-(x) \le f(x) \le f_+(x)$ for all x and $\mathop{\mathbb{E}}_{U}[f_+(U) - f_-(U)] \le \delta$. Then any ε-biased distribution X gives*

$$\left|\mathop{\mathbb{E}}_{X}[f(X)] - \mathop{\mathbb{E}}_{U}[f(U)]\right| \le \delta + \varepsilon \cdot \max\{L(f_+), L(f_-)\}.$$

The functions f_+ and f_- are called sandwiching polynomials for f. This notion of sandwiching is in fact a tight characterisation of small bias [14, Proposition 2.7]. That is, any function f fooled by all small bias generators has sandwiching polynomials satisfying the hypotheses of Proposition 19.

Gopalan et al. [17] use sandwiching polynomials in the analysis of their generator for CNFs. This allows them to set a constant fraction of the bits at each level of recursion ($p = \Omega(1)$), while we set a $1/O(\log n)$ fraction at each level. We would like to similarly exploit sandwiching polynomials for branching programs to improve the seed length of the generator.

A further avenue for improvement would be to modify the generator construction to have $\Theta(1/p)$ levels of recursion, rather than $\Theta(\log(n)/p)$. This would require a significantly different analysis.

References

1 Anil Ada, Omar Fawzi, and Hamed Hatami. Spectral norm of symmetric functions. In *APPROX-RANDOM*, pages 338–349. Springer, 2012.

2 Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost k-wise independent random variables. In *FOCS*, pages 544–553, 1990.

3 Michael Ben-Or and Nathan Linial. Collective coin flipping. *Randomness and Computation*, 5:91–115, 1990.

4 Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width 2 branching programs. *ECCC*, 16:70, 2009.

5 Andrej Bogdanov, Periklis A. Papakonstantinou, and Andrew Wan. Pseudorandomness for read-once formulas. In *FOCS*, pages 240–246, 2011.

6 Jean Bourgain. On the distribution of the fourier spectrum of boolean functions. *Israel J. Mathematics*, 131(1):269–276, 2002.

7 Yigal Brandman, Alon Orlitsky, and John L. Hennessy. A spectral lower bound techniqye for the size of decision trees and two level and/or circuits. *IEEE Transactions on Computers*, 39(2):282–287, 1990.

8 Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *FOCS*, pages 40–47, 2010.

9 Joshua Brody and Elad Verbin. The coin problem and pseudorandomness for branching programs. In *FOCS*, pages 30–39, 2010.

10 Jehoshua Bruck. Harmonic analysis of polynomial threshold functions. *SIAM J. Discrete Mathematics*, 3:168–177, 1990.

11 Jehoshua Bruck and Roman Smolensky. Polynomial threshold functions, AC^0 functions, and spectral norms. *SIAM J. Computing*, 21(1):33–42, 1992.

12 L. Elisa Celis, Omer Reingold, Gil Segev, and Udi Wieder. Balls and bins: Smaller hash families and faster evaluation. In *FOCS*, pages 599–608, 2011.

13 Anindya De. Pseudorandomness for permutation and regular branching programs. In *CCC*, pages 221–231, 2011.

14 Anindya De, Omid Etesami, Luca Trevisan, and Madhur Tulsiani. Improved pseudorandom generators for depth 2 circuits. In Maria Serna, Ronen Shaltiel, Klaus Jansen, and José Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 6302 of *Lecture Notes in Computer Science*, pages 504–517. Springer, 2010.

15 Lars Engebretsen, Piotr Indyk, and Ryan O'Donnell. Derandomized dimensionality reduction with applications. In *SODA*, pages 705–712, 2002.

16 Ehud Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):27–35, 1998.

17 Parikshit Gopalan, Raghu Meka, Omer Reingold, Luca Trevisan, and Salil Vadhan. Better pseudorandom generators from milder pseudorandom restrictions. In *FOCS*, pages 120–129, 2012.

18 Ben Green and Tom Sanders. Boolean functions with small spectral norm. *Geometric and Functional Analysis*, 18(1):144–162, 2008.

19 Vince Grolmusz. On the power of circuits with gates of low ℓ_1 norms. *Theoretical computer science*, 188(1):117–128, 1997.

20 Iftach Haitner, Danny Harnik, and Omer Reingold. On the power of the randomized iterate. In *CRYPTO*, 2006.

21 Alexander Healy, Salil Vadhan, and Emanuele Viola. Using nondeterminism to amplify hardness. *SIAM J. Computing*, 35(4):903–931 (electronic), 2006.

22 R. Impagliazzo, R. Meka, and D. Zuckerman. Pseudorandomness from shrinkage. In *FOCS*, pages 111–119, 2012.

23 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *STOC*, pages 356–364, 1994.

24 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006.

25 Eyal Kaplan, Moni Naor, and Omer Reingold. Derandomized constructions of k-wise (almost) independent permutations. In *APPROX-RANDOM*, pages 354–365, 2005.

26 Michal Koucký, Prajakta Nimbhorkar, and Pavel Pudlák. Pseudorandom generators for group products. In *STOC*, pages 263–272, 2011.

27 Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM J. Computing*, 22(6):1331–1348, 1993.

28 Yishay Mansour. An $O(n \log \log n)$ learning algorithm for DNF under the uniform distribution. *J. CSS*, 50(3):543–550, 1995.

29 Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Computing*, 22:838–856, 1993.

30 Noam Nisan. $\mathcal{RL} \subset \mathcal{SC}$. In *STOC*, pages 619–623, 1992.

31 Noam Nisan and David Zuckerman. More deterministic simulation in logspace. In *STOC*, pages 235–244, 1993.

32 Omer Reingold, Thomas Steinke, and Salil Vadhan. Pseudorandomness for regular branching programs via fourier analysis. In *APPROX-RANDOM*, pages 655–670, 2013.

33 Michael Saks and Shiyu Zhou. $\mathrm{BP_HSPACE}(S) \subset \mathrm{DSPACE}(S^{3/2})$. *J. CSS*, 58(2):376–403, 1999.

34 Amir Shpilka, Avishay Tal, et al. On the structure of boolean functions with small spectral norm. In *ITCS*, pages 37–48, 2014.

35 D. Sivakumar. Algorithmic derandomization via complexity theory. In *CCC*, page 10, 2002.

36 John Steinberger. The distinguishability of product distributions by read-once branching programs. In *CCC*, pages 248–254, 2013.

37 Thomas Steinke. Pseudorandomness for permutation branching programs without the group theory. *ECCC*, 19:83, 2012.

38 Hing Yin Tsang, Chung Hoi Wong, Ning Xie, and Shengyu Zhang. Fourier sparsity, spectral norm, and the log-rank conjecture. In *FOCS*, pages 658–667, 2013.

39 Jiří Šíma and Stanislav Žák. A sufficient condition for sets hitting the class of read-once branching programs of width 3. In *SOFSEM*, pages 406–418, 2012.