Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques

20th International Workshop, APPROX 2017, and
21st International Workshop, RANDOM 2017
August 16–18, 2017, Berkeley, CA, USA

Edited by

Klaus Jansen
José D. P. Rolim
David P. Williamson
Santosh S. Vempala

LIPIcs – Vol. 81 – APPROX/RANDOM 2017 www.dagstuhl.de/lipics

Editors

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

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

Santosh S. Vempala
Georgia Institute of Technology
Georgia, USA
`vempala@gatech.edu`

David P. Williamson
Cornell University
Cornell, USA
`davidpwilliamson@cornell.edu`

ACM Classification 1998
C.2.1 Network Architecture and Design, E.4 Coding and Information Theory, Error Control Codes, F.1.2 Modes of Computation: Online 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, Computations on discrete structures, G.1.2 Approximation, G.2.2 Graph Theory, G.1.3 Numerical Linear Algebra, G.1.6 Optimization, Integer programming, Linear programming, G.2. Discrete Mathematics, G.2.1 Combinatorial algorithms, Counting problems, G.2.2 Graph Theory: Graph algorithms, Network problems, G.3 Probability and Statistics, Probabilistic algorithms (including Monte Carlo), Markov Processes, Statistical computing

ISBN 978-3-95977-044-6

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at http://www.dagstuhl.de/dagpub/978-3-95977-044-6.

Publication date
August, 2017

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.2017.0

ISBN 978-3-95977-044-6 **ISSN 1868-8969** **http://www.dagstuhl.de/lipics**

LIPIcs – Leibniz International Proceedings in Informatics

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

ISSN 1868-8969

http://www.dagstuhl.de/lipics

Contents

Regular Papers

Contributed Talks of APPROX

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Contributed Talks of RANDOM

Preface

This volume contains the papers presented at the 20th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2017) and the 21st International Workshop on Randomization and Computation (RANDOM 2017), which took place concurrently at the at University of California in Berkeley, USA during August 16–18, 2017.

APPROX focuses on algorithmic and complexity issues surrounding the development of efficient approximate solutions to computationally difficult problems, and was the 20th in the series after Aalborg (1998), Berkeley (1999), Saarbrücken (2000), Berkeley (2001), Rome (2002), Princeton (2003), Cambridge (2004), Berkeley (2005), Barcelona (2006), Princeton (2007), Boston (2008), Berkeley (2009), Barcelona (2010), Princeton (2011), Boston (2012), Berkeley (2013), Barcelona (2014), Princeton (2015), and Paris (2016). RANDOM is concerned with applications of randomness to computational and combinatorial problems, and was the 21st workshop in the series following Bologna (1997), Barcelona (1998), Berkeley (1999), Geneva (2000), Berkeley (2001), Harvard (2002), Princeton (2003), Cambridge (2004), Berkeley (2005), Barcelona (2006), Princeton (2007), Boston (2008), Berkeley (2009), Barcelona (2010), Princeton (2011), Boston (2012), Berkeley (2013), Barcelona (2014), Princeton (2015), and Paris (2016).

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 space methods, spectral methods, mathematical programming methods, combinatorial optimization in graphs and networks, algorithmic game theory, mechanism design and economics, computational geometric problems, distributed and parallel approximation, approximate learning, online algorithms, approaches that go beyond worst case analysis, 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, computational learning theory, and other applications of approximation and randomness.

The volume contains 22 contributed papers, selected by the APPROX Program Committee out of 60 submissions, and 27 contributed papers, selected by the RANDOM Program Committee out of 72 submissions.

We would like to thank all the authors who submitted papers, the invited speakers, Uriel Feige and Moses Charikar, the members of the Program Committees, and the external reviewers. We gratefully acknowledge the Department of Computer Science of the Christian-Albrechts-Universität zu Kiel, the Department of Computer Science of the University of Geneva, the College of Computing of the Georgia Institute of Technology, and the School of Operations Research and Information Engineering of the Cornell University.

August 2017

Klaus Jansen, José D. P. Rolim
Santosh S. Vempala, and David P. Williamson

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Organization

Program Committees

APPROX 2017

Nikhil Bansal	Technische Universiteit Eindhoven, The Netherlands
Siu On Chan	The Chinese University of Hong Kong, Hong Kong
Moses Charikar	Stanford University, USA
Michel Goemans	Massachusetts Institute of Technology, USA
Venkatesan Guruswami	Carnegie Mellon University, USA
Sungjin Im	University of California at Merced, USA
Sanjeev Khanna	University of Pennsylvania, USA
Jochen Koenemann	University of Waterloo, Canada
Shi Li	University at Buffalo, USA
Nicole Megow	Universität Bremen, Germany
Viswanath Nagarajan	University of Michigan, USA
Laura Sanità	University of Waterloo, Canada
Ola Svensson	École Polytechnique Fédérale de Lausanne, Switzerland
Seeun William Umboh	Eindhoven University of Technology, The Netherlands
David Williamson (chair)	Cornell University, USA
Anke van Zuylen	College of William & Mary, USA

RANDOM 2017

Shipra Agrawal	Columbia University, USA
Arnab Bhattacharya	Indian Institute of Science, India
Sebastien Bubeck	Microsoft Research, USA
Alan Frieze	Carnegie Mellon University, USA
Anna C. Gilbert	University of Michigan, USA
Thomas Hansen	Aarhus University, Denmark
Anna R. Karlin	University of Washington, USA
Yin Tat Lee	University of Washington, USA
Adam Marcus	Princeton University, USA
Ankur Moitra	Massachusetts Institute of Technology, USA
Richard Peng	Georgia Institute of Technology, USA
Will Perkins	University of Birmingham, United Kingdom
Barna Saha	University of Massachusetts Amherst, USA
Alistair Sinclair	University of California, USA
Santosh Vempala (chair)	Georgia Institute of Technology, USA
David Woodruff	IBM Almaden, USA

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

External Reviewers

Emmanuel Abbe
Ahmad Abdi
Jayadev Acharya
Eric Allender
Sepehr Assadi
Siddharth Barman
Sasha Barvinok
Anna Ben-Hamou
Andre Berger
Antonio Blanca
Olivier Bodini
Trevor Brown
Victor-Emmanuel Brunel
Boris Bukh
Mark Bun
Parinya Chalermsook
Siu Man Chan
Karthekeyan Chandrasekaran
Arkadev Chattopadhyay
Eden Chlamtac
Raphael Clifford
Gil Cohen
Michael B. Cohen
Artur Czumaj
Stephen Desalvo
Ronald de Wolf
Jelena Diakonikolas
Devdatt Dubhashi
Martin Dyer
Ahmed El Alaoui
Marek Elias
Funda Ergun
Moran Feldman
Hendrik Fichtenberger
Nikolaos Fountoulakis
Naveen Garg
Shashwat Garg
Pawel Gawrychowski
Rong Ge
George Giakkoupis
Sivakanth Gopi
Inge Li Gørtz
Catherine Greenhill
Elena Grigorescu
Martin Groß
Heng Guo
Anupam Gupta
Tom Gur
Kristoffer Arnsfelt Hansen
Elad Haramaty
Matan Harel
Nathan Harms
Hamed Hatami
Tyler Helmuth
Kaave Hosseini
Chien-Chung Huang
Sangxia Huang
Lalit Jain
Mark Jerrum
Pritish Kamath
Nathan Keller
Thomas Kesselheim
Yusuke Kobayashi
Swastik Kopparty
Ravishankar Krishnaswamy
Sven Krumke
Janardhan Kulkarni
O-Joung Kwon
Rasmus Kyng
James Lee
Troy Lee
David Levin
Jerry Li
Anita Liebenau
Shachar Lovett
Konstantin Makarychev
Jieming Mao
Jannik Matuschke
Arya Mazumdar
Colin McDiarmid
Or Meir
Benjamin Mirabelli
Ankur Moitra
Tobias Mömke
Meiram Murzabulatov
Cameron Musco
Christopher Musco
Vasileios Nakos
Vishnu Narayan
Amir Nayyeri

Jelani Nelson
Ashkan Norouzi Fard
Kanstantsin Pashkovich
Amelia Perry
Yury Polyanskiy
Ely Porat
Aaron Potechin
Pawel Pralat
Eric Price
Yuri Rabinovich
Miklos Z. Racz
Anup Rao
Ran Raz
Dana Ron
Noga Ron-Zewi
Aaron Roth
Aviad Rubinstein
Atri Rudra
Sushant Sachdeva
Rishi Saket
Rahul Santhanam
Shubhangi Saraf
Ludwig Schmidt
Tselil Schramm
Roy Schwartz
Rocco Servedio
Yanina Shkel
Allan Sly
Aaron Smith
Zhao Song
Daniel Spielman
Aravind Srinivasan
Nikhil Srivastava
He Sun
Ananda Theertha Suresh
Kunal Talwar
Li-Yang Tan
Jakub Tarnawski
Charlotte Truchet
Madhur Tulsiani
Michael Viderman
Thomas Vidick
Marc Vinyals
Junxing Wang
Justin Ward
Osamu Watanabe
Alexander Wein
Omri Weinstein
Andreas Wiese
Ryan Williams
Mary Wootters
Yihong Wu
Lin Yang
Grigory Yaroslavtsev
Anak Yodpinyanee
Joe Yukich
Rico Zenklusen
Peng Zhang
Yuchen Zhang
Baigong Zheng

List of Authors

Rupam Acharyya
Naman Agarwal
Noga Alon
Omer Angel
Itai Ashlagi
Haim Avron
Yossi Azar

Jess Banks
Omri Ben-Eliezer
Anna Ben-Hamou
Kristóf Bérczi
Arnab Bhattacharyya
Vijay Bhattiprolu
Jarosław Błasiok
Glencora Borradaile
Joshua Brakensiek

Sarah Cannon
Marco L. Carmosino
L. Elisa Celis
Karthekeyan Chandrasekaran
Moses Charikar
Xi Chen
Alessandro Chiesa
Ashish Chiplunkar
Kenneth L. Clarkson
Amin Coja-Oghlan

Amit Deshpande
Jian Ding
Dean Doron

Charilaos Efthymiou
Funda Ergün

S. Luna Frank-Fischer
Adam Freilich
Cody R. Freitag
Alan Frieze
Zachary Friggstad

Ofir Geri
Michel X. Goemans
Arnoosh Golestanian
Alexander Golovnev
Sivakanth Gopi
Elena Grigorescu
Anupam Gupta
Venkatesan Guruswami

Samuel Haney
David G. Harris
Chien-Chung Huang

Russell Impagliazzo
Piotr Indyk

Nor Jaafari
Klaus Jansen
Gorav Jindal

Valentine Kabanets
Naonori Kakimura
Sagar Kale
Mihyun Kang
Tobias Kapetanopoulos
Haim Kaplan
Archit Karandikar
Tarun Kathuria
Thomas Kesselheim
Kamyar Khodamoradi
Tamás Király
Kim-Manuel Klein
Robert Kleinberg
Jochen Könemann
Pavel Kolev
Alexandra Kolla
Antonina Kolokolova
Maria Kosche

Leon Ladewig
Lap Chi Lau
Euiwoong Lee
François Le Gall
Vedat Levi Alev
David A. Levin
Ray Li
Edo Liberty

Vivek Madan
Bruce Maggs
Sepideh Mahabadi
Biswaroop Maiti
Rahul Makhijani
Peter Manohar
Christopher Martin
Abbas Mehrabian
Cristopher Moore

Jelani Nelson

Maciej Obremski
Neil Olver

Debmalya Panigrahi
Kanstantsin Pashkovich
Wesley Pegden
Richard Peng
Thomas Pensyl
Yuval Peres
Eric Price

Yuval Rabani
Mirmahdi Rahgoshay
Rajmohan Rajaraman
R. Ravi
Oded Regev
Mohsen Rezapour
Tim Roughgarden
Ronitt Rubinfeld

Erfan Sadeqi Azer
Mohammad R. Salavatipour
Saurabh Sawlani
Rocco A. Servedio
Igor Shinkar
Maciej Skorski
Aravind Srinivasan
Alexandre Stauffer
Daniel Štefankovič
Damian Straszak
Timothy Sun
Ravi Sundaram
Maxim Sviridenko
Chaitanya Swamy
William J. Swartworth

Li-Yang Tan
Avishay Tal
Inbal Talgam-Cohen
Amnon Ta-Shma
Sumedh Tirodkar
Andreas Tönnis
Khoa Trinh

Jonathan Ullman
Francisco Unda

Ali Vakilian
Ameya Velingker
Santhoshini Velusamy
Rakesh Venkat
Nisheeth K. Vishnoi
Ilya Volkovich
Jan Vondrák
Jens Vygen

Erik Waingarten
Yuyi Wang
Thomas Watson
Roger Wattenhofer
Omri Weinstein
David P. Woodruff
Mary Wootters

Chao Xu

Anak Yodpinyanee
Yuichi Yoshida

Yifeng Zhang
Baigong Zheng
Samson Zhou

Min-Cost Bipartite Perfect Matching with Delays*

Itai Ashlagi[1], Yossi Azar[2], Moses Charikar[3], Ashish Chiplunkar[4], Ofir Geri[5], Haim Kaplan[6], Rahul Makhijani[7], Yuyi Wang[8], and Roger Wattenhofer[9]

1 Department of Management Science and Engineering, Stanford University, Stanford, CA, USA
iashlagi@stanford.edu

2 School of Computer Science, Tel Aviv University, Tel Aviv, Israel
azar@tau.ac.il

3 Department of Computer Science, Stanford University, Stanford, CA, USA
moses@cs.stanford.edu

4 School of Computer Science, Tel Aviv University, Tel Aviv, Israel
ashish.chiplunkar@gmail.com

5 Department of Computer Science, Stanford University, Stanford, CA, USA
ofirgeri@cs.stanford.edu

6 School of Computer Science, Tel Aviv University, Tel Aviv, Israel
haimk@post.tau.ac.il

7 Departmentof Management Science and Engineering, Stanford University, Stanford, CA, USA
rahulmj@stanford.edu

8 Department of Information Technology and Electrical Engineering, ETH Zürich, Zürich, Switzerland
yuwang@ethz.ch

9 Department of Information Technology and Electrical Engineering, ETH Zürich, Zürich, Switzerland
wattenhofer@ethz.ch

Abstract

In the min-cost bipartite perfect matching with delays (MBPMD) problem, requests arrive online at points of a finite metric space. Each request is either positive or negative and has to be matched to a request of opposite polarity. As opposed to traditional online matching problems, the algorithm does not have to serve requests as they arrive, and may choose to match them later at a cost. Our objective is to minimize the sum of the distances between matched pairs of requests (the connection cost) and the sum of the waiting times of the requests (the delay cost). This objective exhibits a natural tradeoff between minimizing the distances and the cost of waiting for better matches. This tradeoff appears in many real-life scenarios, notably, ride-sharing platforms. MBPMD is related to its non-bipartite variant, min-cost perfect matching with delays (MPMD), in which each request can be matched to any other request. MPMD was introduced by Emek et al. (STOC'16), who showed an $O(\log^2 n + \log \Delta)$-competitive randomized algorithm on n-point metric spaces with aspect ratio Δ.

Our contribution is threefold. First, we present a new lower bound construction for MPMD and MBPMD. We get a lower bound of $\Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)$ on the competitive ratio of any randomized algorithm for MBPMD. For MPMD, we improve the lower bound from $\Omega(\sqrt{\log n})$ (shown by Azar et al., SODA'17) to $\Omega\left(\frac{\log n}{\log\log n}\right)$, thus, almost matching their upper bound of $O(\log n)$. Second, we adapt the algorithm of Emek et al. to the bipartite case, and provide a simplified analysis

* Yossi Azar and Ashish Chiplunkar were supported in part by the Israel Science Foundation (grant no. 1506/16), by the I-CORE program (center no. 4/11), and by the Blavatnik Fund.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 1; pp. 1:1–1:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

that improves the competitive ratio to $O(\log n)$. The key ingredient of the algorithm is an $O(h)$-competitive randomized algorithm for MBPMD on weighted trees of height h. Third, we provide an $O(h)$-competitive deterministic algorithm for MBPMD on weighted trees of height h. This algorithm is obtained by adapting the algorithm for MPMD by Azar et al. to the apparently more complicated bipartite setting.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity, F.1.2 [Modes of Computation] Online Computation

Keywords and phrases online algorithms with delayed service, bipartite matching, competitive analysis

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.1

1 Introduction

In many marketplaces agents arrive and are matched over time. Examples include matching drivers with passengers in ride-sharing platforms, matching players in online games, forming exchanges between patient-donor incompatible pairs in kidney exchange, and even labor markets. While agents in such marketplaces are interested in high quality matches, waiting is costly. This paper studies the problem of matching in a centralized marketplace, in which agents' preferences are induced by the "distance" to their potential matches.

Consider, for example, a ride-sharing platform that is faced with the problem of matching passengers with drivers who arrive to different locations at different times, and assume that a passenger can match with any available driver. A greedy approach would match each arriving passenger upon arrival (if possible) to the closest available driver. This approach can be, however, very inefficient; for instance, imagine a passenger at location x is matched with a driver at location y and only seconds later a driver becomes available at location x. In particular, there is a natural tradeoff between making the market thicker in order to form better matches and the costs it imposes on waiting agents.

We explore this tradeoff in an online setting with no information about the arriving agents (worst-case input). Allowing agents to wait is a key property that differentiates this paper from traditional online matching, where agents are to be matched upon arrival. This idea of delayed service in the context of matching has been introduced recently by Emek et al. [12]. There is also a growing body of work on dynamic matching problems with delays under stochastic assumptions [2, 3, 4, 6]. The concept of delayed service is also relevant for other online problems with and without stochastic assumptions.

In the problem we study, requests arrive in an online manner at the n points of a finite metric space. Each request is identified by its time of arrival, its location, and its polarity, which can be either positive or negative (multiple requests may arrive at the same location). Each request can only be matched with a request of the opposite polarity. In the motivating example, requests correspond to the drivers and passengers, who arrive at different times and different locations, and the polarities of requests imply that passengers can only match with drivers and vice versa. The objective of the social planner is to minimize the sum of the *delay cost*, which is the time since the arrival of each request until it is matched, and the *connection cost*, which is the sum of distances between each two requests that are matched to each other.

We call this problem *min-cost bipartite perfect matching with delays* (MBPMD), as the requests can be represented by a bipartite graph with edge weights that correspond to the distances in the metric space. We measure the performance of a matching algorithm using the notion of competitive ratio: an algorithm is α-competitive if for every input, the cost incurred by the algorithm is at most α times the cost of the optimal solution.

MBPMD is an extension of the problem studied by Emek et al. [12], min-cost perfect matching with delays (MPMD), in which all requests are of the same type and each can be matched with any other request. Emek et al. provided a randomized algorithm with competitive ratio $O(\log^2 n + \log \Delta)$ using probabilistic embedding into tree metrics, where Δ is the ratio of the maximum distance to the minimum distance in the metric space. MPMD was studied further by Azar et al. [5], who recently found an algorithm that improved the competitive ratio to $O(\log n)$, and established a lower bound of $\Omega\left(\sqrt{\log n}\right)$.

The bipartite version of MPMD is more natural in many applications: matching in ride-sharing platforms, job markets, and any situation where there are two types of entities that need to be matched to each other. While MPMD and MBPMD seem quite similar, there is no clear reduction from MBPMD to MPMD, and the study of MBPMD results in a more technically involved analysis. For example, an issue that arises only in MBPMD is that there can be many requests with the same polarity waiting at the same location without being able to match. In contrast, any reasonable algorithm for MPMD will have at most one request waiting at each location, as it would immediately match requests that are waiting at the same location.

Our Contribution

Our contribution has three parts. First, we present a new lower bound construction for MPMD and MBPMD. We show that in a metric space containing n equally spaced points in the unit interval, the competitive ratio of any randomized algorithm for MBPMD is at least $\Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)$. Our construction also provides a lower bound of $\Omega\left(\frac{\log n}{\log\log n}\right)$ on the competitive ratio of any randomized algorithm for MPMD, which improves the current lower bound of $\Omega(\sqrt{\log n})$ shown by Azar et al. [5], and matches their upper bound up to the $\log\log n$ factor.

Second, we adapt the randomized algorithm of Emek et al. [12] to MBPMD and provide a considerably simplified analysis that results in a competitive ratio of $O(\log n)$. Our analysis can be applied to the non-bipartite case as well. At a high-level, at any time the algorithm computes a tentative matching that pairs requests in a greedy manner. Two paired requests are matched after waiting a time drawn from an exponential random variable with mean that equals the distance between them.

The randomized algorithm consists of a preprocessing phase, in which the finite metric space is embedded into a tree metric, and of a randomized greedy algorithm that solves MBPMD on tree metrics. Informally, we say that an algorithm A is (β, γ)-competitive if for every benchmark algorithm A^*, the (expected) cost incurred by A is at most β times the connection cost of A^* plus γ times the delay cost of A^*. Our analysis shows that the randomized greedy algorithm is $(3, 6h+1)$-competitive, where h is the height of the tree.

Using probabilistic embedding into HSTs [8, 9, 14] and the height reduction step of Bansal et al. [7], any finite metric space can be embedded into a tree metric with height $O(\log n)$ and expected distortion $O(\log n)$. Using this embedding, we can turn any $(O(1), O(h))$-competitive algorithm for tree metrics into a $O(\log n)$-competitive algorithm for any finite metric space.

The third part of our contribution is a *deterministic* $(10, 10h)$-competitive algorithm for MBPMD on tree metrics. This algorithm is an adaptation of the MPMD algorithm by Azar et al. [5] in the sense that both algorithms buy the edges required to connect the requests by paying in installments, and connect two requests as soon as all the edges on the path between them have been bought. The algorithm for MPMD pays, at a uniform rate, for an edge if the number of requests waiting at the leaves under it is odd. In our case, we pay for an edge if the numbers of positive and negative requests under it are unequal, and the rates at which we pay for edges are non-uniform. The rate of payment for an edge is proportional to the magnitude of imbalance between positive and negative requests under the edge. This introduces a substantial amount of complication in the analysis, and to mitigate it, it becomes inevitable to use the technique of potential functions. We remark that this deterministic algorithm gives rise to a *barely random* algorithm for MBPMD on general metrics, that is, the number of random bits it uses is independent of the size of the online input.

Related Work

Most related to this work are the papers by Emek et al. [12] and Azar et al. [5], who studied min-cost matching with delays (MPMD) (i.e., the non-bipartite case). As mentioned above, Emek et al. [12] introduced the notion of online problems with delayed service and provided an $O\left(\log^2 n + \log \Delta\right)$-competitive algorithm for MPMD. Azar et al. [5] provided an $O(h)$-competitive deterministic algorithm for MPMD on tree metrics, and used it to show an $O(\log n)$-competitive algorithm for general metrics. Additionally, they provided a lower bound of $\Omega\left(\sqrt{\log n}\right)$ on the competitive ratio of randomized algorithms for MPMD.

Another strand of research in the economics and operations literatures studied matching with delays in stochastic and more structural environments. Anderson et al. [3] and Ashlagi et al. [4] study a model with an underlying stochastic graph and assume agents arrive according to some process. They seek to minimize agents' average waiting time and find that greedy matching is asymptotically optimal. Akbarpour et al. [2] allow for agents departures and find that when departure times are known, greedy matching leads to a suboptimal match rate. These papers do not have the notion of distance; agents only care about when they match and not whom they match to, which is key to the fact that greedy matching performs well. Baccara et al. [6] look at a two-sided market where on each side agents can be of one of two types and one type is of higher "quality" than the other. They assume a single agent on each side arrives every time period and find that the optimal matching policy accumulates agents up to a certain threshold.

Online bipartite matching, in general, is an extremely popular model. In the original problem studied by Karp et al. [15], vertices on one side of a bipartite graph are known in advance and vertices on the other side arrive online. Each vertex on the online side can match to only some of the offline vertices, and can only match *upon arrival.* The goal is to maximize the number of matched vertices. There are many extensions and variants of this problem: maximum vertex-weighted matching [1, 11], the AdWords problem [17], and others. The literature on online matching is extensive; see [16] for a survey.

2 Preliminaries

A *metric space* $\mathcal{M}$ is a set S equipped with a distance function $d : S \times S \longrightarrow \mathbb{R}^{\geq 0}$ such that $d(x, y) = 0$ if and only if $x = y$, $d(x, y) = d(y, x)$ for all $x, y \in S$, and $d(x, y) + d(y, z) \geq d(x, z)$ for all $x, y, z \in S$. The problem of min-cost bipartite perfect matching with delays (MBPMD) is an online problem defined on an underlying finite metric space $\mathcal{M} = (S, d)$ as follows. An online input instance I over S is a sequence of requests $\langle(p_i, b_i, t_i)\rangle_{i=1}^{m}$, where p_i is a point in the metric space, $b_i \in \{+1, -1\}$, and t_i is the time at which the request arrives. We assume that the number of *positive* requests ($b_i = +1$) equals the number of *negative* requests ($b_i = -1$). The algorithm is required to output a perfect matching between the positive requests and the negative requests. In min-cost perfect matching with delays (MPMD), requests do not have polarity: each request can match to any other request and the algorithm is required to output a perfect matching (we assume that the total number of requests is even). For each pair (i, j) of requests output by the algorithm at time t (where $t \geq \max(t_i, t_j)$), the algorithm pays a connection cost of $d(p_i, p_j)$ and a delay cost of $(t - t_i) + (t - t_j)$. The offline connection cost of creating the pair (i, j) is $d(p_i, p_j)$, and the offline delay cost is $|t_i - t_j|$.

3 The Lower Bounds

The focus of this section is to prove the following lower bound results.

▶ **Theorem 1.** *There is an n-point metric space on which any randomized algorithm for MPMD has competitive ratio $\Omega(\log n/ \log\log n)$ against an oblivious adversary.*

▶ **Theorem 2.** *There is an n-point metric space on which any randomized algorithm for MBPMD has competitive ratio $\Omega(\sqrt{\log n/ \log\log n})$ against an oblivious adversary.*

To prove a lower bound of α on the competitive ratio of randomized online algorithms, we use Yao's min-max principle [10, 18, 19], and give a distribution over input instances which defeats every deterministic algorithm by the factor α. The required distributions for proving the above two lower bounds are very similar; a random MBPMD instance is generated by generating a random MPMD instance and giving polarities to the requests in a randomized fashion. Due to the inherent similarity, we merge the descriptions of the two distributions.

The metric space is given by a parameter L, which is an even integer. Let $n = 2L^{\left\lfloor \frac{L}{\log_2 L} \right\rfloor} \leq 2^{L+1}$, so that $L = \Theta(\log n)$. The required metric space consists of n equally spaced points on the real interval $[0, 1]$. All asymptotic notation in this section is with respect to $n \to \infty$, or equivalently $L \to \infty$.

Every instance in the support of the distribution consists of requests given in $r = \lfloor L/ \log_2 L \rfloor$ phases. In each phase, the requests are given at once at the beginning, and they are equally spaced in $[0, 1]$. Furthermore, the set of points at which these requests are given is a suitably chosen random subset of the set of points at which requests were given in the previous phase. The number of requests in phase i is $n_i = 2L^{r-i}$, and the duration of phase i is $t_i = 1/L^{r-i}$ time units. The distribution $\mathcal{D}$ on M(B)PMD instances is generated as follows.

- Let $r := \lfloor L/ \log_2 L \rfloor$, $n := 2L^r$, $S_0 := \{1/n, 2/n, \ldots, 1\}$.
- For $i = 0, \ldots, r$,
  1. **Only for MBPMD:** Choose b_i uniformly at random from $\{+1, -1\}$.
  2. Give requests at points in S_i. **Only for MBPMD:** Starting with the polarity b_i for the leftmost request in S_i, assign alternating polarities to the requests.
  3. Index the points in S_i from left to right, with index 1 for the leftmost point. Construct sets $Y_0^{i+1}, Y_1^{i+1} \subseteq S_i$ as follows.
     a. Y_0^{i+1} is the set of points whose index is an integer multiple of L.
     b. Y_1^{i+1} is the set of points whose index is an integer multiple of L plus $L/2$, that is, $L/2$, $3L/2$, and so on. (Recall that L is even.)
  4. Choose z_{i+1} uniformly at random from $\{0, 1\}$. Let $S_{i+1} := Y_{z_{i+1}}^{i+1}$. (Thus, $|S_{i+1}| = |Y_0^{i+1}| = |Y_1^{i+1}| = |S_i|/L$.)
  5. Wait for time $t_i = 1/L^{r-i}$ (and then move on to the next phase, if $i < r$).

In order to bound the expected cost of an arbitrary deterministic M(B)PMD algorithm, we need to set up some notation and prove a key lemma. For a set S of requests on an underlying metric space, and a real number c, let $\mathrm{MIN}(S, c)$ denote the cost of the min-cost (possibly partial) matching on S (ignoring signs, even in MBPMD), where the cost of a matching is the sum of distances between the matched pairs of requests, plus a penalty of c per unmatched request. The following lemma can be thought of as a triangle inequality on sets of requests.

▶ **Lemma 3.** *Let X, Y_0, and Y_1 be arbitrary sets of requests on an underlying metric space. Then* $\mathrm{MIN}(X \cup Y_0, c) + \mathrm{MIN}(X \cup Y_1, c) \geq \mathrm{MIN}(Y_0 \cup Y_1, c)$.

Proof. For $j \in \{0, 1\}$, let M_j be the matching on the set $X \cup Y_j$ which achieves the cost $\mathrm{MIN}(X \cup Y_j, c)$. Consider the set of edges $M_0 \cup M_1$. This is a union of vertex-disjoint paths in which every vertex in $Y_0 \cup Y_1$ has degree at most one. Thus, each path has all its vertices, except possibly the endpoints, in X.

Construct a matching M on $Y_0 \cup Y_1$ as follows. For each maximal path p in $M_0 \cup M_1$, do the following. If both endpoints of p are in X (which means that the whole path p is in X), ignore p. Else if p has length 0, that is, p is a single vertex from $Y_0 \cup Y_1$, leave it unmatched in M. Else, denote the endpoints of p by u and v. If both u and v are in $Y_0 \cup Y_1$, match u and v in M, and charge this cost to the weight of p. If $u \in X$ and $v \in Y_0 \cup Y_1$, leave v unmatched in M, and charge this cost to the cost of leaving u unmatched in one of the M_js. Thus, the contribution of every path to the cost $\mathrm{MIN}(X \cup Y_0, c) + \mathrm{MIN}(X \cup Y_1, c)$ is at least as much as its contribution to the cost of M. ◀

We use the above lemma to prove the following lower bound on the cost of an arbitrary deterministic online M(B)PMD algorithm in every phase.

▶ **Lemma 4.** *Every deterministic online M(B)PMD algorithm incurs a cost of at least $1/4$ in expectation in every phase i, conditioned on $z_1, \ldots, z_{i-1}$ (and $b_0, \ldots, b_{i-1}$, additionally, for MBPMD).*

Proof. Let X be the set of pending requests from earlier stages at the beginning of an arbitrary phase i. If we condition on the random events from the previous phases, X is fixed. Recall that t_i, the duration of the phase, is $1/L^{r-i}$. Since no new requests arrive while the phase is in progress, we may assume that each request which is matched during the phase is matched at the beginning of the phase. Thus, each unmatched request waits from the beginning till the end of the phase, resulting in a delay cost of t_i. Hence, the expected cost of the algorithm is at least $\mathbb{E}_{z_i}[\mathrm{MIN}(X \cup S_i, t_i)] = (\mathrm{MIN}(X \cup Y_0^i, t_i) + \mathrm{MIN}(X \cup Y_1^i, t_i))/2$. (This holds even in the case of MBPMD, because $\mathrm{MIN}(X \cup Y_j^i, t_i)$ is the cost of the best possible matching ignoring polarities, whereas the algorithm produces a matching which respects polarities and can only have a larger cost.) Thus, by Lemma 3, the algorithm's expected cost is bounded from below by $\mathrm{MIN}(Y_0^i \cup Y_1^i, t_i)/2$.

Observe that $Y_0^i \cup Y_1^i$ is a set of $4L^{r-i}$ equispaced requests with spacing $1/4L^{r-i}$. Thus, $\mathrm{MIN}(Y_0^i \cup Y_1^i, t_i) = \mathrm{MIN}(Y_0^i \cup Y_1^i, 1/L^{r-i}) = 1/2$, since it is cheaper to match all requests in $Y_0^i \cup Y_1^i$ and pay $1/8L^{r-i}$ per request, rather than paying $1/L^{r-i}$ per unmatched request. Therefore, the cost of the algorithm is at least $\mathrm{MIN}(Y_0^i \cup Y_1^i, t_i)/2 \geq 1/4$ in every phase i. ◀

▶ **Corollary 5.** *Every deterministic online M(B)PMD algorithm incurs a cost of at least $r/4 = \Omega(L/\log L)$ in expectation on a random input drawn from $\mathcal{D}$.*

Proof. Follows by unconditioning the bound from Lemma 4. ◀

Next, we construct offline solutions to the instances of M(B)PMD drawn from $\mathcal{D}$, thereby giving upper bounds on the cost of the optimum solution.

▶ **Lemma 6.** *Every MPMD instance generated from $\mathcal{D}$ has a solution of cost at most $1 + 2/\log_2 L = O(1)$.*

Proof. Construct a solution as follows. For i decreasing from r to 1, connect each unmatched request from phase i to the request from phase $i-1$ located at the same point. This is possible because $S_i \subseteq S_{i-1}$, and results in zero connection cost. The delay cost is at most the number of requests in phase i times the duration of phase $i-1$, which is $2L^{r-i} \cdot 1/L^{r-i+1} = 2/L$. Finally pair up the unmatched requests from phase 0 optimally, with connection cost at most 1. The overall connection cost is 1, and the overall delay cost is 2/L for each phase except phase 0. Thus, the total cost of the solution is $1 + 2r/L \leq 1 + 2/\log_2 L$. ◀

▶ **Lemma 7.** *The expected cost of the optimal solution of an MBPMD instance generated from the distribution $\mathcal{D}$ is $O(\sqrt{L/\log L})$.*

For proving Lemma 7, we need some notation. Fix an instance of MBPMD in the support of $\mathcal{D}$. For $x \in [0,1]$, define the *phase-i cumulative surplus* at x to be the signed total of the requests from phase i that are located in $[0,x]$, and denote it by $\operatorname{csur}_i(x)$. Then $\operatorname{csur}_i(x) \in \{0,1\}$ if $b_i = +1$, and $\operatorname{csur}_i(x) \in \{-1,0\}$ if $b_i = -1$. Define $\operatorname{csur}(x) = \sum_{i=0}^{r} \operatorname{csur}_i(x)$, the *cumulative surplus* at x, which is the signed total of all requests from all phases that are located in $[0,x]$. Observe that for any x, any feasible solution to the instance must connect at least $|\operatorname{csur}(x)|$ requests located to the left of x to the same number of requests located to the right of x. Hence, the connection cost of any feasible solution must be at least $\int_0^1 |\operatorname{csur}(x)|dx$. Moreover, there exists a solution, say SOL, whose connection cost is precisely $\int_0^1 |\operatorname{csur}(x)|dx$ (connect the t^{th} positive request and the t^{th} negative request from the left, for all t). This will be our adversarial solution to the instance. In order to bound the connection cost of SOL from above, we need prove that $\mathbb{E}_{b_0,\dots,b_r,z_1,\dots,z_r}[\int_0^1 |\operatorname{csur}(x)|dx]$ is small. We prove something stronger: we prove that the expectation is small enough even if we condition over the values of $z_1,\dots,z_r$, and only average over $b_0,\dots,b_r$.

▶ **Lemma 8.** *For every fixed $(z_1,\dots,z_r) \in \{0,1\}^r$, $\mathbb{E}_{b_0,\dots,b_r}\left[\int_0^1 |\operatorname{csur}(x)|dx\right] = O(\sqrt{r})$.*

Proof. Since $\mathbb{E}_{b_0,\dots,b_r}\left[\int_0^1 |\operatorname{csur}(x)|dx\right] = \int_0^1 \mathbb{E}_{b_0,\dots,b_r}\left[|\operatorname{csur}(x)|\right]dx$, it is sufficient to prove that $\mathbb{E}_{b_0,\dots,b_r}\left[|\operatorname{csur}(x)|\right] = O\left(\sqrt{r}\right)$ for every $x \in [0,1]$.

Given $z_1,\dots,z_r$, the locations of the requests are fixed. Observe that $\operatorname{csur}_i(x)$ is zero if the number of requests of phase i in $[0,x]$ is even. If that number is odd, then $\operatorname{csur}_i(x) = b_i$ is $+1$ and -1 with probability 1/2 each. Thus, $\operatorname{csur}(x) = \sum_{i=0}^{r} \operatorname{csur}_i(x)$ is the sum of at most $r+1$ independent random variables, each of which takes values $+1$ and -1 with equal probability, where the number of random variables is determined by x and $z_1,\dots,z_r$. Therefore $|\operatorname{csur}(x)|$ is the deviation of a random walk of at most $r+1$ steps on the integers starting from 0, and moving in either direction with equal probability. Using a standard result,[1] we have $\mathbb{E}\left[|\operatorname{csur}(x)|\right] = O(\sqrt{r})$, as required. ◀

Taking the solution SOL which minimizes the connection cost as the adversarial solution, we now prove an upper bound on the expected cost of the optimum solution of a random MBPMD instance drawn from $\mathcal{D}$.

[1] For instance: http://mathworld.wolfram.com/RandomWalk1-Dimensional.html.

Proof of Lemma 7. Consider the solution SOL. By Lemma 8, its expected connection cost is $O(\sqrt{r}) = O(\sqrt{L/\log L})$, and we are left to bound its expected delay cost. Note that the sum of the arrival times of all requests in an instance is an upper bound on the delay cost of every solution to the instance (which keeps a request waiting only until its partner arrives). In particular, this applies to SOL. For instances in the support of $\mathcal{D}$, the sum of the arrival times is the same, and is equal to $\sum_{i=0}^{r} n_i \sum_{j=0}^{i-1} t_j$, where $n_i = 2L^{r-i}$ is the number of requests in phase i, and $t_j = 1/L^{r-j}$ is the duration of phase j. Thus, the delay cost is bounded from above by

$$\sum_{i=0}^{r} n_i \sum_{j=0}^{i-1} t_j = \sum_{i=0}^{r} 2L^{r-i} \sum_{j=0}^{i-1} \frac{1}{L^{r-j}} = 2\sum_{i=0}^{r} \frac{1}{L^i} \sum_{j=0}^{i-1} L^j = 2\sum_{i=0}^{r} \frac{1}{L^i} \cdot \frac{L^i - 1}{L-1} \leq \frac{2r}{L-1}$$

which is $O(1/\log L)$, since $r = \lfloor L/\log L \rfloor$. Thus, the expected cost of a random MBPMD instance drawn from $\mathcal{D}$ is $O(\sqrt{L/\log L}) + O(1/\log L) = O(\sqrt{L/\log L})$. ◀

Finally, we use the lower bound on the algorithm's cost and the upper bounds on the optimum cost to prove lower bounds on the competitive ratio of MPMD and MBPMD.

Proof of Theorem 1. Follows from Corollary 5 and Lemma 6. ◀

Proof of Theorem 2. Follows from Corollary 5 and Lemma 7. ◀

4 The $O(\log n)$ Upper Bound for MBPMD: Overview

Our focus in this section is to give an algorithm for MBPMD on arbitrary metrics, and thus, to prove the following result.

▶ **Theorem 9.** *There exists a randomized online algorithm with a competitive ratio of $O(\log n)$ for MBPMD on n-point metric spaces.*

As stated previously, we establish the above theorem by reducing MBPMD on arbitrary metrics to MBPMD on tree metrics. A tree metric is given by a tree with positive edge weights such that the points of the metric are the vertices of the tree and the distance between two points is the length of the simple path connecting them. To achieve the reduction, we use the following result (Lemma 3.1 of [5]), which is an easy consequence of probabilistic embedding into tree metrics [14] and Lemma 5.1 of [7].

▶ **Lemma 10.** *Any n-point metric space $\mathcal{M}$ can be embedded, with distortion $O(\log n)$, into a distribution $\mathcal{D}$ supported on metrics induced by trees of height $O(\log n)$.*

Informally, the *distortion* of an embedding is an upper bound on the expected blowup in the distances between pairs of points.

Analogous to Azar et al. [5], we use the more general notion of (β, γ)-*competitiveness* in addition to the usual notion of competitive ratio. Reusing their notation, given an instance I of MBPMD and an arbitrary solution SOL of I, we let SOL_d denote its connection cost with respect to the metric d, SOL_t denote its delay cost, and (with a slight abuse of notation) SOL denote its total cost. We restate the definition of (β, γ)-competitiveness for the sake of completeness.

▶ **Definition 11.** Given a randomized online algorithm $\mathcal{A}$ for MBPMD on a metric space $\mathcal{M} = (S, d)$ and an instance I on S, $\mathcal{A}(I)$ denotes the expected cost of $\mathcal{A}$ on I. $\mathcal{A}$ is said to be α-*competitive* if for every I and every solution SOL of I, $\mathcal{A}(I) \leq \alpha \cdot \text{SOL}$. $\mathcal{A}$ is said to be (β, γ)-*competitive* if for every I and every solution SOL of I, $\mathcal{A}(I) \leq \beta \cdot \text{SOL}_d + \gamma \cdot \text{SOL}_t$.

Given an embedding of a metric space into another with distortion μ, and a (β, γ)-competitive algorithm for the embedding metric, it is easy to see that it can be turned into a $(\mu\beta, \mu\gamma)$-competitive algorithm for the original metric. However, Emek et al. [12] observed that this can strengthened slightly to the following lemma, whose proof is deferred to Appendix A.

▶ **Lemma 12.** *Suppose that a metric space $\mathcal{M} = (S, d)$ can be embedded into a distribution $\mathcal{D}$ supported on metric spaces over a set $S' \supseteq S$ with distortion μ. Additionally, suppose that for every metric space $\mathcal{M}'$ in the support of $\mathcal{D}$, there is a (possibly randomized) online (β, γ)-competitive algorithm $\mathcal{A}^{\mathcal{M}'}$ for MBPMD on $\mathcal{M}'$. Then there is a $(\mu\beta, \gamma)$-competitive (and thus, $(\max(\mu\beta, \gamma))$-competitive) algorithm $\mathcal{A}$ for MBPMD on $\mathcal{M}$.*

In the next two sections, we give two online algorithms for MBPMD, both of which are $(O(1), O(h))$-competitive on edge-weighted trees of height h. Theorem 9 then follows easily.

Proof of Theorem 9. Given an n-point metric space $\mathcal{M}$, using Lemma 10, embed it into a distribution $\mathcal{D}$ over metrics given by edge-weighted trees of height $O(\log n)$ with distortion $O(\log n)$. The algorithms for tree metrics from the next two sections are $(O(1), O(\log n))$-competitive for every tree metric in the support of $\mathcal{D}$ (Theorems 13 and 17). Therefore, by Lemma 12, there is an $O(\log n)$-competitive algorithm for MBPMD on every metric $\mathcal{M}$. ◀

Notation

We state here notation that will be used in the description and analysis of the algorithms. Suppose the tree metric is given by an edge-weighted tree T rooted at an arbitrary vertex r. For a vertex u, let T_u denote the maximal subtree of T rooted at u, e_u denote the edge between u and its parent, and d_u denote the weight of e_u (d_r is defined to be infinity). Similarly, for $e = e_u$ we also use T_e to denote T_u (the subtree rooted at the lower endpoint of e). Let h be the height of the tree, that is, the maximum of the number of vertices in the path between r and any leaf. We assume, without loss of generality, that the requests are given only at the leaves of T. (If not, we pretend as if each non-leaf vertex u has a child u' at distance zero, which is a leaf, and the requests are given at u' instead of u.) Let $\text{lca}(u, v)$ denote the lowest common ancestor of vertices u and v in the tree. Given an edge-weighted tree T, rooted at vertex r, and a set of requests on the leaves of T, we define the *surplus* of a vertex v to be the number of positive requests minus the number of negative requests in T_v, and denote it by $\text{sur}(v)$. (Note that $\text{sur}(v)$ can be negative.) While comparing the performance of the algorithm with a candidate solution SOL, we use $\text{sur}^*(v)$ to denote the surplus of v when running SOL. We also use $\text{sur}(e)$ and $\text{sur}^*(e)$ to denote the surplus of an edge e. If $e = e_u$, then $\text{sur}(e) = \text{sur}(u)$ and $\text{sur}^*(e) = \text{sur}^*(u)$.

5 A Randomized Algorithm for MBPMD on Trees

In this section, we adapt the randomized algorithm for MPMD on trees presented by Emek et al. [12] to the bipartite case. We present a simplified analysis which shows that the algorithm is $(3, 6h + 1)$-competitive. The original analysis was restricted to binary hierarchically well-separated trees, and together with the embedding step resulted in a competitive ratio of $O\left(\log^2 n + \log \Delta\right)$ for general metrics. By lifting the binary HST restriction and using the embedding method of Lemma 10, our analysis improves the competitive ratio to $O(\log n)$. The algorithm appears here as Algorithm 1.

Algorithm 1 A Randomized Algorithm for MBPMD on Tree Metrics

Greedy matching (computed upon each arrival): While there are two unmatched requests of opposite polarities at the same point, match those requests immediately. For all other requests, compute a *tentative* greedy matching as follows:

- Consider the vertices from the leaves to the root. (Formally, choose any order such that each vertex is considered only after all of its children have already been considered.)
- When considering a vertex v, let P be the set of positive requests in T_v that are not tentatively matched yet, and N be the set of negative requests in T_v that are not tentatively matched yet.
- While P and N are both non-empty, tentatively match a request from P to a request from N and remove the requests from P and N. Break ties arbitrarily.

At each infinitesimally small time step $[t, t + dt)$: For each two requests p_1, p_2 that are tentatively matched, match these two requests with probability $\frac{dt}{d(p_1,p_2)}$ where $d(p_1, p_2)$ is the length of the simple path between p_1 and p_2 in the tree. In that case, we say that the match is realized.

▶ Remark. Algorithm 1 is described in terms of infinitesimally small discrete time steps. However, it can be also described continuously as follows. For each two requests p_1, p_2 that are tentatively matched, that match will be realized after waiting a time period of Z where $Z \sim Exp\left(\frac{1}{d(p_1,p_2)}\right)$.

▶ **Theorem 13.** *Algorithm 1 for MBPMD on tree metrics is* $(3, 6h + 1)$*-competitive, and hence,* $(6h + 1)$*-competitive.*

The proof of Theorem 13 has two parts. First, we bound the connection cost of Algorithm 1 in terms of the connection and delay costs of any benchmark algorithm SOL (Lemma 14). Second, we show how to bound the delay cost of the algorithm using the connection cost of the algorithm and the delay cost SOL_t (Lemma 16).

We introduce some notation used in the proof. Let $T = (V, E)$ be a tree with weight function $w : E \to R^{>0}$ (defining a tree metric (V, d)). Denote the connection cost of Algorithm 1 on (V, d) by ALG_d, the delay cost by ALG_t, the total cost by ALG, and let SOL be any benchmark solution for MBPMD on the same tree metric. Let $\mathrm{ALG}_d(t_1, t_2)$ denote the connection cost of the algorithm only due to matches that occur in the time interval $[t_1, t_2)$, and $\mathrm{ALG}_t(t_1, t_2)$ denote the delay cost incurred by ALG during that time interval. $\mathrm{SOL}_d(t_1, t_2)$ and $\mathrm{SOL}_t(t_1, t_2)$ are defined similarly.

For each edge $e \in E$, let $P_e(t), N_e(t)$ denote the number of unmatched positive and negative requests (respectively) inside T_e in ALG at time t, and define $P^*_e(t), N^*_e(t)$ similarly for SOL. Using these definitions, at time t, $\mathrm{sur}(e) = P_e(t) - N_e(t)$ and $\mathrm{sur}^*(e) = P^*_e(t) - N^*_e(t)$.

We remark on a few properties of the algorithm. First, each edge e can be used as part of at most $|\mathrm{sur}(e)|$ matches at time t. It may be used for less than $|\mathrm{sur}(e)|$ matches, e.g., if there are not enough requests that can be matched to those waiting in T_e. Second, all the requests from T_e that are matched through e are of the same polarity (otherwise, they would have been matched at a lower level). With these observations, we are ready to prove the key lemma of the section.

▶ **Lemma 14.** $\mathbb{E}[\mathrm{ALG}_d] \leq \mathrm{SOL}_d + 2h \cdot \mathrm{SOL}_t$

Proof. For each edge $e \in E$, define the following potential at time t:

$$\Phi_e(t) = w(e)\,|\mathrm{sur}(e) - \mathrm{sur}^*(e)| = w(e)\,|P_e(t) - N_e(t) - (P^*_e(t) - N^*_e(t))|$$

The total potential at time t is defined as $\Phi(t) = \sum_{e \in E} \Phi_e(t)$.

We divide the time into intervals. The first interval starts at time 0. An interval ends and the next interval begins when a new request arrives or when SOL matches two requests. Let $[t_1, t_2)$ be an interval and denote $\Delta\Phi = \Phi(t_2) - \Phi(t_1)$. We wish to prove that

$$\mathbb{E}[\mathrm{ALG}_d(t_1, t_2) + \Delta\Phi] \leq \mathrm{SOL}_d(t_1, t_2) + 2h \cdot \mathrm{SOL}_t(t_1, t_2).$$

There are three events that can happen: the arrival of a request, a match by SOL, or a match by ALG. Arrivals and matches by SOL can only happen at time t_1 during the interval. Matches by ALG can happen at any time in (t_1, t_2) (the probability that a match occurs at time t_1 is 0).

Arrival. We claim that the arrival of a request does not change the potential. For each edge e in the path from the request to the root, either P_e and P_e^* or N_e and N_e^* increase by 1, and Φ_e remains the same. If ALG or SOL matches the new request to another request at the same location, the surplus (and also the potential) does not change.

Match by SOL. The connection cost $\mathrm{ALG}_d(t_1, t_2)$ is not affected by the actions of SOL. Note that the connection cost incurred by SOL due to a single match is the sum of weights of edges that are used as part of the match, and that the match only changes the potential of these edges. For each edge e that is used as part of a match, $\Delta\Phi_e \leq w(e)$. By summing over all these edges, we get $\Delta\Phi \leq \mathrm{SOL}_d$.

Match by ALG. At any time $t \in (t_1, t_2)$, there are no arrivals or matches in SOL. Hence, the tentative matching maintained by the algorithm does not change, and the potential can only change due to a match by ALG. If a match of a pair of requests is realized by ALG, the potential of each edge e in the path connecting these two requests either increases or decreases by $w(e)$.

Let e be an edge that is used in the tentative matching, and denote by $\mathrm{ALG}_e(t, t')$ the connection cost that ALG incurred during (t, t') due to edge e. The following claim relates the expected connection cost and change in potential at edge e to the surplus in SOL and to the length of the interval $[t_1, t_2)$, which we will relate to the delay cost of SOL. Note that $|\mathrm{sur}^*(e)|$ does not change during (t_1, t_2) (as there are no arrivals or matches in SOL). The proof is deferred to Appendix B.

▶ **Claim 15.** *For every edge e that is used in the tentative matching,* $\mathbb{E}[\mathrm{ALG}_e(t_1, t_2) + \Delta\Phi_e] \leq 2\,|\mathrm{sur}^*(e)|\,(t_2 - t_1)$.

The claim asserts that the expected connection cost due to the use of e and the change in Φ_e is at most $2\,|\mathrm{sur}^*(e)| \cdot (t_2 - t_1)$. Now we claim that there are least $|\mathrm{sur}^*(e)|$ requests waiting in SOL in the subtree of e. This follows from the fact that $\max\{P_e^*(t), N_e^*(t)\} \geq |\mathrm{sur}^*(e)|$. The delay cost $\mathrm{SOL}_t(t_1, t_2)$ due to these requests is $|\mathrm{sur}^*(e)|\,(t_2 - t_1)$.

We have shown that for every edge e, we can "charge" the sum of the expected connection cost incurred by ALG and the change in potential to the requests waiting in SOL in T_e: this sum is at most $2\,|\mathrm{sur}^*(e)|\,(t_2 - t_1)$, while there are at least $|\mathrm{sur}^*(e)|$ requests waiting in SOL each leading to a delay cost of $t_2 - t_1$. A request waiting in SOL is charged at most once for each edge on the path that connects the request to the root of the tree, that is, each request is charged at most h times.

Summing over all the edges, we get that during the interval $[t_1, t_2)$,

$$\mathbb{E}[\mathrm{ALG}_d(t_1, t_2) + \Delta\Phi] \leq \mathrm{SOL}_d(t_1, t_2) + 2h \cdot \mathrm{SOL}_t(t_1, t_2).$$

Algorithm 2 A Deterministic Algorithm for MBPMD on Tree Metrics

```
Initialize: $F^+ := \emptyset$, $F^- := \emptyset$. For each vertex $u$, $z_u^+ := 0$ and $z_u^- := 0$.
At every moment:
- While there are two unmatched requests of opposite polarities at the same point, match those requests immediately.
- For each vertex $u$, if $u$ is positively unsaturated and $\mathrm{sur}(u) > 0$ (resp. $u$ is negatively unsaturated and $\mathrm{sur}(u) < 0$), then increase counter $z_u^+$ (resp. $z_u^-$) at the rate $\mathrm{sur}(u)$ (resp. $-\mathrm{sur}(u)$). Else, keep the counter frozen.
- For each vertex $u \neq r$, as soon as the value of $z_u^+$ (resp. $z_u^-$) becomes equal to $2d_u$, add the edge $e_u$ to $F^+$ (resp. $F^-$). This makes $u$ positively saturated (resp. negatively saturated), and $z_u^+$ (resp. $z_u^-$) is frozen.
- For each positive request, located at $u$, and each negative request, located at $v$, as soon as the entire path between $u$ and $\mathrm{lca}(u, v)$ is contained in $F^+$, and the entire path between $v$ and $\mathrm{lca}(u, v)$ is contained in $F^-$,
    - Connect the request at $u$ to the request at $v$.
    - Remove the edges on the path from $u$ to $v$ from both $F^+$ as well as $F^-$.
    - For every vertex $w \neq \mathrm{lca}(u, v)$ on the path from $u$ to $v$, reset $z_w^+ := 0$ and $z_w^- := 0$. (All these vertices are unsaturated due to the previous step.)
```

The lemma follows by summing these inequalities for all intervals and by noticing that the potential is 0 at time 0 and after both algorithms have matched all the requests. ◀

The following lemma is similar to Lemma 7 in [12] (Lemma 4.8 in the full version [13]). The proof is deferred to Appendix B.

▶ **Lemma 16.** $\mathbb{E}[\mathrm{ALG}_t] \leq 2\mathbb{E}[\mathrm{ALG}_d] + \mathrm{SOL}_t$

Proof of Theorem 13. From Lemma 16, we get

$$\mathbb{E}[\mathrm{ALG}] = \mathbb{E}[\mathrm{ALG}_t] + \mathbb{E}[\mathrm{ALG}_d] \leq 3\mathbb{E}[\mathrm{ALG}_d] + \mathrm{SOL}_t\,.$$

By Lemma 14, $\mathbb{E}[\mathrm{ALG}_d] \leq \mathrm{SOL}_d + 2h \cdot \mathrm{SOL}_t$. We conclude that $\mathbb{E}[\mathrm{ALG}] \leq 3\,\mathrm{SOL}_d + (6h + 1)\,\mathrm{SOL}_t$. ◀

6 A Deterministic Algorithm for MBPMD on Trees

The algorithm, which appears here as Algorithm 2, maintains two forests, F^+ and F^-, both initialized to be empty. For every vertex u, the algorithm also maintains two counters, z_u^+ and z_u^-, initially set to zero. Intuitively, if $e_u \in F^+$ (resp. $e_u \in F^-$), then e_u is available for connecting a positive (resp. negative) request inside T_u to a negative (resp. positive) request outside. We say that a vertex u is *positively saturated* (resp. *negatively saturated*) if the edge e_u is in F^+ (resp. F^-), else, we say it is *positively unsaturated* (resp. *negatively unsaturated*). The root r is always positively as well as negatively unsaturated, by definition. Note F^+ and F^- are not necessarily disjoint, and therefore, a vertex can be both positively as well as negatively saturated at the same time. The rest of the section is dedicated to proving the following theorem.

▶ **Theorem 17.** *Algorithm 2 for MBPMD on tree metrics is* $(10, 10h)$*-competitive, and hence,* $10h$*-competitive.*

For any vertex u, we divide time into phases as follows. The first phase at u starts when the algorithm starts. Whenever the edge e_u is used to connect requests, the phase at u ends and a new phase begins at u. Note that the last phase at any u is necessarily incomplete, and that the phases at different vertices need not be aligned. Observe that at the beginning of any phase at u, both z_u^+ and z_u^- are zero, whereas at the end, one of them is equal to $2d_u$ and the other is at most $2d_u$.

For the analysis, imagine a variable y_u^+ (resp. y_u^-) for every u, which increases at the same rate as z_u^+ (resp. z_u^-) during the run of the algorithm, but which is never reset to zero. We will separately relate the connection cost as well as the delay cost of the algorithm to $\sum_u (y_u^+ + y_u^-)$, and then relate $\sum_u (y_u^+ + y_u^-)$ to the cost of an arbitrary solution SOL, and thus, prove $(O(1), O(h))$-competitiveness.

▶ **Lemma 18.** *The connection cost of the algorithm is at most* $\frac{1}{2}\sum_u (y_u^+ + y_u^-)$.

Proof. For an arbitrary vertex u, recall that every usage of edge e_u, which results in a connection cost of d_u, marks the end of a phase at u. In every phase at u, one of z_u^+ and z_u^- increases from 0 to $2d_u$. Thus, in every phase at u, $y_u^+ + y_u^-$ increases by at least $2d_u$. This implies the claim. ◀

▶ **Lemma 19.** *The delay cost of the algorithm is at most* $2\sum_u (y_u^+ + y_u^-)$.

We defer the proof of the above lemma to Appendix C. Now we need to relate the value $\sum_u (y_u^+ + y_u^-)$ at the end of the algorithm's run to the cost of an arbitrary solution SOL to the instance. For this, let x_u be the total delay cost incurred by SOL due to requests inside T_u, and x'_u be the total connection cost incurred by SOL for using the edge e_u.

▶ **Lemma 20.** *At the end of the algorithm's run, for all vertices* u, $y_u^+ + y_u^- \leq 4(x_u + x'_u)$.

We defer the proof to Appendix C. Next, we relate $\sum_u (x_u + x'_u)$ to the cost of the solution SOL. Denoting the distance function of the tree metric by d, recall that SOL_d and SOL_t denote the connection cost and the delay cost of SOL, respectively. Our final ingredient is Lemma 3.6 from [5], stated as follows.

▶ **Lemma 21.** $\sum_u (x_u + x'_u) \leq \mathrm{SOL}_d + h \cdot \mathrm{SOL}_t$.

The competitiveness of the algorithm now follows easily.

Proof of Theorem 17. From Lemmas 18 and 19, the algorithm's total cost is at most $\frac{5}{2}\sum_u (y_u^+ + y_u^-)$. By Lemma 20, this is at most $10\sum_u (x_u + x'_u)$, which by Lemma 21, is at most $10\,\mathrm{SOL}_d + 10h \cdot \mathrm{SOL}_t$. Therefore, the algorithm is $(10, 10h)$-competitive. ◀

7 Concluding Remarks and Open Problems

In this paper, we showed a randomized $O(\log n)$-competitive algorithm and a lower bound of $\Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)$ on the competitive ratio of any randomized algorithm for MBPMD. One natural open problem is closing the gap between these bounds. Another open question is whether randomization is needed in solving MBPMD. While for trees we provided a deterministic $O(h)$-competitive algorithm, the question of finding deterministic algorithms or lower bounds for general metrics remains open.

We took here the centralized planner's view that can dictate who can match to whom. An interesting open question is what is the efficiency loss if the market is decentralized and agents selfishly decide whether to match with a partner or to wait for a closer partner.

There are some modeling decisions to make here, but in general the competitive ratio should increase, since agents will impose negative externalities on others (such analysis is done under stochastic assumptions in [6]).

Our model only scratches the surface of the numerous variants of MBPMD that can be practical for many applications. Keeping the ride-sharing motivating example in mind, one can model carpooling as a many-to-one matching problem while taking into account the different destinations of the passengers. One can also allow requests to move to other points while waiting, simulating drivers that head toward busy areas while waiting for a match.

Further study of MBPMD can involve different assumptions on the input. While we analyzed the competitive ratio for worst-case input, a more refined analysis can be made in the case where the input is drawn from some known distribution. Another possible analysis beyond the worst case is to consider a more restricted family of metric spaces with a structure that may result in better bounds even for worst-case input.

Finally, the delay of services and allocations shows up in many applications, which can be studied using the notion of online problems with delayed service.

Acknowledgments. We thank Amos Fiat for his insightful involvement in the discussions.

References

1 Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. Online vertex-weighted bipartite matching and single-bid budgeted allocations. In *Proceedings of the Twenty-second Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1253–1264, 2011. doi:10.1137/1.9781611973082.95.

2 Mohammad Akbarpour, Shengwu Li, and Shayan Oveis Gharan. Thickness and information in dynamic matching markets. *Available at SSRN 2394319*, 2017.

3 Ross Anderson, Itai Ashlagi, David Gamarnik, and Yash Kanoria. A dynamic model of barter exchange. In *Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1925–1933, 2015. doi:10.1137/1.9781611973730.129.

4 Itai Ashlagi, Maximilien Burq, Patrick Jaillet, and Vahideh H. Manshadi. On matching and thickness in heterogeneous dynamic markets. In *Proceedings of the 2016 ACM Conference on Economics and Computation*, page 765, 2016. doi:10.1145/2940716.2940758.

5 Yossi Azar, Ashish Chiplunkar, and Haim Kaplan. Polylogarithmic bounds on the competitiveness of min-cost perfect matching with delays. In *Proceedings of the Twenty-eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1051–1061, 2017. doi:10.1137/1.9781611974782.67.

6 Mariagiovanna Baccara, SangMok Lee, and Leeat Yariv. Optimal dynamic matching. *Available at SSRN 2641670*, 2015.

7 Nikhil Bansal, Niv Buchbinder, Aleksander Madry, and Joseph Naor. A polylogarithmic-competitive algorithm for the k-server problem. *J. ACM*, 62(5):40, 2015. doi:10.1145/2783434.

8 Yair Bartal. Probabilistic approximations of metric spaces and its algorithmic applications. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, pages 184–193, 1996. doi:10.1109/SFCS.1996.548477.

9 Yair Bartal. Graph decomposition lemmas and their role in metric embedding methods. In *Proceedings of the Twelfth Annual European Symposium on Algorithms*, pages 89–97, 2004. doi:10.1007/978-3-540-30140-0_10.

10 Allan Borodin and Ran El-Yaniv. On randomization in on-line computation. *Inf. Comput.*, 150(2):244–267, 1999. doi:10.1006/inco.1998.2775.

11 Nikhil R. Devanur, Kamal Jain, and Robert D. Kleinberg. Randomized primal-dual analysis of RANKING for online bipartite matching. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 101–107, 2013. `doi:10.1137/1.9781611973105.7`.

12 Yuval Emek, Shay Kutten, and Roger Wattenhofer. Online matching: haste makes waste! In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 333–344, 2016. `doi:10.1145/2897518.2897557`.

13 Yuval Emek, Shay Kutten, and Roger Wattenhofer. Online matching: Haste makes waste!, 2016. arXiv:1603.03024 [cs.DS]. URL: `http://arxiv.org/abs/1603.03024`.

14 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004. `doi:10.1016/j.jcss.2004.04.011`.

15 Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 352–358, 1990. `doi:10.1145/100216.100262`.

16 Aranyak Mehta. Online matching and ad allocation. *Found. Trends Theor. Comput. Sci.*, 8(4):265–368, October 2013. `doi:10.1561/0400000057`.

17 Aranyak Mehta, Amin Saberi, Umesh V. Vazirani, and Vijay V. Vazirani. Adwords and generalized online matching. *J. ACM*, 54(5), 2007. `doi:10.1145/1284320.1284321`.

18 Leen Stougie and Arjen P. A. Vestjens. Randomized algorithms for on-line scheduling problems: how low can't you go? *Oper. Res. Lett.*, 30(2):89–96, 2002. `doi:10.1016/S0167-6377(01)00115-8`.

19 Andrew Chi-Chih Yao. Probabilistic computations: Toward a unified measure of complexity (extended abstract). In *18th Annual Symposium on Foundations of Computer Science*, pages 222–227, 1977. `doi:10.1109/SFCS.1977.24`.

A Proof Omitted from Section 4

Recall the following definition of a metric embedding and its distortion.

▶ **Definition 22.** Let $\mathcal{M} = (S, d)$ be a finite metric space, and let $\mathcal{D}$ be a probability distribution over metrics on a finite set $S' \supseteq S$. We say that $\mathcal{M}$ *embeds* into $\mathcal{D}$ with *distortion* μ if

- For every $x, y \in S$ and every metric space $\mathcal{M}' = (S', d')$ in the support of $\mathcal{D}$, we have $d(x, y) \leq d'(x, y)$.
- For every $x, y \in S$, we have $\mathbb{E}_{\mathcal{M}'=(S',d')\sim\mathcal{D}}[d'(x, y)] \leq \mu \cdot d(x, y)$.

Proof of Lemma 12. Algorithm $\mathcal{A}$ simply samples a metric space $\mathcal{M}' = (S', d')$ from the distribution $\mathcal{D}$, and simulates the behavior of $\mathcal{A}^{\mathcal{M}'}$. Clearly, the delay cost paid by $\mathcal{A}$ is the same as the delay cost paid by $\mathcal{A}^{\mathcal{M}'}$. Furthermore, since $d(p, q) \leq d'(p, q)$ for all $p, q \in S$, the connection cost paid by $\mathcal{A}$ is no more than the connection cost paid by $\mathcal{A}^{\mathcal{M}'}$. Fix an input instance I of MBPMD on $\mathcal{M}$, and an arbitrary solution SOL of I. Then the expected cost $\mathcal{A}(I)$ of the algorithm $\mathcal{A}$ on I is bounded as follows.

$$\mathcal{A}(I) \leq \mathbb{E}_{\mathcal{M}'=(S',d')\sim\mathcal{D}}[\mathcal{A}^{\mathcal{M}'}(I)]$$

where $\mathcal{A}^{\mathcal{M}'}(I)$ is the expected cost of $\mathcal{A}^{\mathcal{M}'}$ on I, the expectation being taken over the randomness internal to $\mathcal{A}^{\mathcal{M}'}$. Since $\mathcal{A}^{\mathcal{M}'}$ is (β, γ)-competitive, we have by definition,

$$\mathcal{A}^{\mathcal{M}'}(I) \leq \beta \, \mathrm{SOL}_{d'} + \gamma \, \mathrm{SOL}_t$$

This implies

$$\begin{aligned}\mathcal{A}(I) &\leq \mathbb{E}_{\mathcal{M}'=(S',d')\sim\mathcal{D}}[\beta\,\mathrm{SOL}_{d'}+\gamma\,\mathrm{SOL}_t]\\ &= \beta\cdot\mathbb{E}_{\mathcal{M}'=(S',d')\sim\mathcal{D}}[\mathrm{SOL}_{d'}]+\gamma\,\mathrm{SOL}_t\\ &\leq \beta\mu\cdot\mathrm{SOL}_d+\gamma\cdot\mathrm{SOL}_t\end{aligned}$$

where the equality follows from linearity of expectation and the fact that SOL_t is independent of $\mathcal{M}'$, while the second inequality follows from the definition of distortion. By the definition of (β,γ)-competitiveness, the claim follows. ◀

B Proofs Omitted from Section 5

Proof of Claim 15. Consider an edge e that is used for (tentatively) matching k positive requests $p_1,\ldots,p_k$ to k negative requests $n_1,\ldots,n_k$. Remember that all the requests in T_e that are matched through e are of the same type. Assume that the requests in T_e are the positive requests $p_1,\ldots,p_k$, that is, for every match that is realized, P_e decreases by 1 (the case where the requests under T_e are negative is analogous). There are two cases.

Case 1: $\mathrm{sur}^*(e)\leq 0$. In that case, since $\mathrm{sur}(e)$ decreases by 1 for each match that is realized, the potential Φ_e decreases by $w(e)$ for each match that ALG makes. Thus, for each match the connection cost incurred by ALG for using edge e and the change in the potential sum to 0. Namely, $\mathbb{E}[\mathrm{ALG}_e(t_1,t_2)+\Delta\Phi_e]=0\leq 2\,|\mathrm{sur}^*(e)|\,(t_2-t_1)$.

Case 2: $\mathrm{sur}^*(e)>0$. In that case, as long as $\mathrm{sur}(e)>\mathrm{sur}^*(e)$, each match will decrease Φ_e by $w(e)$, and when $\mathrm{sur}(e)\leq\mathrm{sur}^*(e)$, each match will increase Φ_e by $w(e)$. Note that at time t_1, $\mathrm{sur}(e)\geq k>0$. Let $t'\in[t_1,t_2)$ be the minimal time such that $\mathrm{sur}(e)\leq\mathrm{sur}^*(e)$. If there is no such t', we set $t'=t_2$.

For $1\leq i\leq k$, let X_i be an indicator random variable for the event that the request p_i is matched during the interval (t',t_2) (conditioned on p_i not being matched before t'), and Z_i be an exponential random variable with parameter $\frac{1}{d(p_i,n_i)}$. Then,

$$\mathbb{E}[X_i]=\Pr[Z_i<t_2|Z_i>t]=\Pr[Z_i<t_2-t']=1-e^{-\frac{t_2-t'}{d(p_i,n_i)}}\leq 1-e^{-\frac{t_2-t_1}{w(e)}}\leq\frac{t_2-t_1}{w(e)}.$$

Now, note that

$$\begin{aligned}\mathbb{E}[\mathrm{ALG}_e(t_1,t_2)+\Delta\Phi_e] &= \mathbb{E}[\mathrm{ALG}_e(t_1,t')+\Phi_e(t')-\Phi_e(t_1)]\\ &\quad+\mathbb{E}[\mathrm{ALG}_e(t',t_2)+\Phi_e(t_2)-\Phi_e(t')]\\ &= \mathbb{E}[\mathrm{ALG}_e(t_1,t')+\Phi_e(t')-\Phi_e(t_1)]\\ &\quad+\mathbb{E}[\mathrm{ALG}_e(t',t_2)+\Phi_e(t_2)-\Phi_e(t')|t'<t_2]\Pr[t'<t_2]\\ &\leq \mathbb{E}[\mathrm{ALG}_e(t_1,t')+\Phi_e(t')-\Phi_e(t_1)]\\ &\quad+\mathbb{E}[\mathrm{ALG}_e(t',t_2)+\Phi_e(t_2)-\Phi_e(t')|t'<t_2]\end{aligned}$$

where we use the facts that $\mathbb{E}[\mathrm{ALG}_e(t',t_2)+\Phi_e(t_2)-\Phi_e(t')|t'=t_2]=0$ and that $\mathrm{ALG}_e(t',t_2)+\Phi_e(t_2)-\Phi_e(t')$ is non-negative (the potential can only increase due to matches in (t',t_2)).

If ALG makes N_1 matches during (t_1,t'), then $\mathrm{ALG}_e(t_1,t')=N_1\cdot w(e)$, while $\Phi_e(t')-\Phi_e(t_1)=-N_1\cdot w(e)$ (before t', the potential only decreases due to the matches). Thus, $\mathbb{E}[\mathrm{ALG}_e(t_1,t')+\Phi_e(t')-\Phi_e(t_1)]=0$.

Consider $\mathbb{E}[\mathrm{ALG}_e(t', t_2) + \Phi_e(t_2) - \Phi_e(t') | t' < t_2]$. Note that at if $t' < t_2$, then $N_1 = \max\{0, \mathrm{sur}(e) - \mathrm{sur}^*(e)\}$.[2] Then, during the interval (t', t_2), there are $k - N_1 \leq |\mathrm{sur}^*(e)|$ requests that may be matched using edge e. Denote the number of requests that are matched using e during (t', t_2) by N_2. Intuitively, $\mathbb{E}[N_2]$ is at most $(k - N_1) \cdot \frac{t_2 - t_1}{w(e)}$, since for each such request i, $\mathbb{E}[X_i] \leq \frac{t_2 - t_1}{w(e)}$.

Formally, if for $S \subseteq \{1, \ldots, k\}$ of size N_1, A_S denotes the event that the requests $\{p_i | i \in S\}$ are matched before t',

$$\begin{aligned}
\mathbb{E}[N_2 | t' < t_2] &= \sum_{\substack{S \subseteq \{1,\ldots,k\}: \\ |S| = N_1}} \mathbb{E}[N_2 | t' < t_2, A_S] \Pr[A_S | t' < t_2] \\
&= \sum_{\substack{S \subseteq \{1,\ldots,k\}: \\ |S| = N_1}} \left(\sum_{i \notin S} \mathbb{E}[X_i | t' < t_2, A_S] \right) \Pr[A_S | t' < t_2] \\
&\leq \sum_{\substack{S \subseteq \{1,\ldots,k\}: \\ |S| = N_1}} \left(\sum_{i \notin S} \frac{t_2 - t_1}{w(e)} \right) \Pr[A_S | t' < t_2] \\
&= (k - N_1) \cdot \frac{t_2 - t_1}{w(e)} \sum_{\substack{S \subseteq \{1,\ldots,k\}: \\ |S| = N_1}} \Pr[A_S | t' < t_2] \\
&= (k - N_1) \cdot \frac{t_2 - t_1}{w(e)} \\
&\leq |\mathrm{sur}^*(e)| \cdot \frac{t_2 - t_1}{w(e)}
\end{aligned}$$

Finally, since for each request matched after t', the potential increases by $w(e)$, we get that

$$\begin{aligned}
\mathbb{E}[\mathrm{ALG}_e(t', t_2) + \Phi_e(t_2) - \Phi_e(t') | t' < t_2] &= 2w(e) \cdot \mathbb{E}[N_2 | t' < t_2] \\
&\leq |\mathrm{sur}^*(e)| \cdot 2w(e) \cdot \frac{t_2 - t_1}{w(e)} \\
&= 2\, |\mathrm{sur}^*(e)|\, (t_2 - t_1)
\end{aligned}$$

and $\mathbb{E}[\mathrm{ALG}_e(t_1, t_2) + \Delta\Phi_e] \leq 2\, |\mathrm{sur}^*(e)|\, (t_2 - t_1)$. ◀

Proof of Lemma 16. We divide the time into intervals as in the proof of Lemma 14. The first interval starts at time 0. An interval ends and the next interval begins when a new request arrives or when SOL matches two requests. Let $[t_1, t_2)$ be an interval. We show that $\mathbb{E}[\mathrm{ALG}_t(t_1, t_2)] \leq 2\mathbb{E}[\mathrm{ALG}_d(t_1, t_2)] + \mathrm{SOL}_t(t_1, t_2)$.

Note that the number of requests that are not tentatively matched at time t is $|\mathrm{sur}(r)|$, and that at any time t, $\mathrm{sur}(r) = \mathrm{sur}^*(r)$ (both ALG and SOL run on the same input and clear requests in pairs of different types). Intuitively, since SOL also had a surplus of the same number of requests, the delay cost incurred by SOL is at least the delay cost incurred due to the requests that are not tentatively matched by ALG. Formally, note that $\mathrm{sur}(r)$ has the same value at all times $t \in (t_1, t_2)$. Let $K = |\mathrm{sur}(r)|$ for some $t \in (t_1, t_2)$. Then, the delay

[2] This refers to $\mathrm{sur}(e)$ at time t_1. Note that at most one match is realized at time t' (the probability that two matches will be realized at the same time is 0). Therefore, N_1 must be $\max\{0, \mathrm{sur}(e) - \mathrm{sur}^*(e)\}$ and not greater than that.

cost of the requests that are not tentatively matched during $[t_1, t_2)$ is $K \cdot (t_2 - t_1)$ (note that the tentative matching cannot not be recomputed in the middle of an interval). In addition, during (t_1, t_2), SOL has at least K requests waiting, hence $\mathrm{SOL}_t(t_1, t_2) \geq K \cdot (t_2 - t_1)$.

So far we have shown that during $[t_1, t_2)$, the delay cost of the requests that are not tentatively matched is at most $\mathrm{SOL}_t(t_1, t_2)$. We now consider all the requests that are part of the tentative matching computed by ALG. For each pair of requests, we compare the expected connection cost and expected delay cost. Let p_1, p_2 be two requests that are tentatively matched by ALG. We denote the connection cost due to p_1, p_2 during $[t_1, t_2)$ by $\Delta\,\mathrm{ALG}_d(p_1, p_2)$ and the delay cost due to p_1, p_2 during $[t_1, t_2)$ by $\Delta\,\mathrm{ALG}_t(p_1, p_2)$.

The time until p_1, p_2 are matched is an exponential random variable Z with parameter $\frac{1}{d(p_1,p_2)}$. The requests are matched during the interval if $Z < t_2 - t_1$. Then, the expected connection cost is $\mathbb{E}[\Delta\,\mathrm{ALG}_d(p_1, p_2)] = d(p_1, p_2) \cdot \Pr[Z < t_2 - t_1] = d(p_1, p_2)(1 - e^{-\frac{t_2 - t_1}{d(p_1,p_2)}})$. The expected delay cost for each one of p_1, p_2 during $[t_1, t_2)$ is

$$\mathbb{E}[min\{Z, t_2 - t_1\}] = \mathbb{E}[Z - (Z - (t_2 - t_1))^+] = d(p_1, p_2)(1 - e^{-\frac{t_2 - t_1}{d(p_1,p_2)}}).$$

Since both p_1 and p_2 wait, we get that

$$\mathbb{E}[\Delta\,\mathrm{ALG}_t(p_1, p_2)] = 2d(p_1, p_2)(1 - e^{-\frac{t_2 - t_1}{d(p_1,p_2)}}) = 2\mathbb{E}[\Delta\,\mathrm{ALG}_d(p_1, p_2)].$$

Summing over all the pairs of tentatively matched requests and adding the delay cost of the unmatched requests, we get that

$$\mathbb{E}[\mathrm{ALG}_t(t_1, t_2)] \leq 2\mathbb{E}[\mathrm{ALG}_d(t_1, t_2)] + \mathrm{SOL}_t(t_1, t_2).$$

We conclude the proof of the lemma by summing over all the intervals, and by linearity of expectation, we get

$$\mathbb{E}[\mathrm{ALG}_t] \leq 2\mathbb{E}[\mathrm{ALG}_d] + \mathrm{SOL}_t\,. \qquad ◀$$

C Proofs Omitted from Section 6

In order to prove Lemma 19, we need the following observation.

▶ **Observation 23.** *Given a set of requests on the vertices of T which contains an equal number of positive and negative requests, let M be a minimum cost perfect matching between the positive and the negative requests (where, as usual, the cost of matching two requests is the distance between their locations under the tree metric). Then for any vertex v of the tree, the number of requests inside T_v that are matched in M to requests outside T_v is precisely $|\,\mathrm{sur}(v)|$. Furthermore, all these requests have the same sign as $\mathrm{sur}(v)$.*

Proof of Lemma 19. Let U^+ (resp. U^-) denote the set of positively (resp. negatively) unsaturated vertices v with $\mathrm{sur}(v) > 0$ (resp. $\mathrm{sur}(v) < 0$). Note that U^+ and U^- are disjoint. From the description of the algorithm, the rate of increase of $\sum_u (y_u^+ + y_u^-)$ is $\sum_{v \in U^+ \cup U^-} |\,\mathrm{sur}(v)|$. The rate of increase of the delay cost is equal to the number of pending requests. Thus, it is sufficient to prove that the number of pending requests is at most $2\sum_{v \in U^+ \cup U^-} |\,\mathrm{sur}(v)|$ at any time (except at instants when requests are connected).

Consider an arbitrary time instant. Recall that $\mathrm{sur}(r)$ is equal to the number of positive pending requests minus the number of negative pending requests. If $\mathrm{sur}(r) \neq 0$, augment the set of pending requests with $|\,\mathrm{sur}(r)|$ artificial requests located at r, with sign opposite to the sign of $\mathrm{sur}(r)$, resulting in a balanced set of requests. Let M be a minimum cost perfect matching between the positive and the negative requests in this set. First, consider the $|\operatorname{sur}(r)|$ pending requests that get matched to the $|\operatorname{sur}(r)|$ augmented requests at r. Charge these pending requests to r, and note that $r \in U^+ \cup U^-$ (unless $\operatorname{sur}(r) = 0$). Next, let (R^+, R^-) be a match in M, where R^+ (resp. R^-) is a positive (resp. negative) pending request located at u^+ (resp. u^-), and let $v = \operatorname{lca}(u^+, u^-)$. Since the algorithm has not connected R^+ and R^-, at least one of the following must be true.

1. There is a vertex $v' \neq v$ on the path from u^+ to v such that $e_{v'} \notin F^+$, i.e. v' is positively unsaturated.
2. There is a vertex $v' \neq v$ on the path from u^- to v such that $e_{v'} \notin F^-$, i.e. v' is negatively unsaturated.

Consider the first case. By Observation 23, since M matches the positive request $R^+ \in T_{v'}$ to $R^- \notin T_{v'}$, we have $\operatorname{sur}(v') > 0$. Additionally, since v' is positively unsaturated, $v' \in U^+$. By similar argument, in the second case, $v' \in U^-$. In either case, charge the pair of requests (R^+, R^-) to the vertex $v' \in U^+ \cup U^-$.[3] Observe that if this charging scheme charges a pair of pending requests to a vertex v, then one of the requests is in T_v, the other is outside T_v, and the pair is included in M. Again, by Observation 23, the number of pairs charged to any vertex v is at most $|\operatorname{sur}(v)|$. Thus, the number of pending requests is at most $2\sum_{v \in U^+ \cup U^-} |\operatorname{sur}(v)|$, as required. ◀

Proof of Lemma 20. We use the potential function technique. We design a potential function ϕ such that in each phase, the changes $\Delta(y_u^+ + y_u^-)$, $\Delta\phi$, and $\Delta(x_u + x'_u)$ satisfy

$$\Delta(y_u^+ + y_u^-) + \Delta\phi \leq 4\Delta(x_u + x'_u) \tag{1}$$

and $\phi = 0$ at the beginning as well as at the end of the algorithm's run. Summing (1) over all phases, we get the result.

Recall that $\operatorname{sur}^*(u)$ denotes the surplus of vertex u resulting from SOL. Define $\phi = 4d_u \cdot |\operatorname{sur}^*(u) - \operatorname{sur}(u)|$. Clearly, at the beginning as well as at the end, we have $\operatorname{sur}(u) = \operatorname{sur}^*(u) = 0$, and thus, $\phi = 0$. Observe that $\operatorname{sur}^*(u) - \operatorname{sur}(u)$ (and hence, ϕ) remains unchanged when new requests are given. The only events resulting in a change in $\operatorname{sur}^*(u) - \operatorname{sur}(u)$ are either SOL or the algorithm connecting a request inside T_u to one outside T_u. Also, x_u increases at a rate of at least $|\operatorname{sur}^*(u)|$.

In each phase of a vertex u, each of y_u^+ and y_u^- increases by at most $2d_u$, and therefore, $\Delta(y_u^+ + y_u^-) \leq 4d_u$. Except the last phase, in every phase, at least one of y_u^+ and y_u^- increases by exactly $2d_u$, and the phase ends with the algorithm connecting a request inside T_u to one outside T_u. We call such a phase *complete*, and we call the last phase *incomplete*. We prove that (1) holds first for complete phases, and then for the incomplete phase.

Let $k \geq 0$ denote the (absolute) number of requests in T_u which SOL connected to requests outside T_u during an arbitrary phase. Thus, $\Delta x'_u \geq k d_u$.

Consider any complete phase of vertex u and, without loss of generality, assume that the phase ends due to a positive request inside T_u getting connected to a negative request outside T_u. This means that z_u^+ increases from 0 to $2d_u$ in the phase. Since the only events resulting in a change in $\operatorname{sur}^*(u) - \operatorname{sur}(u)$ are either SOL or the algorithm connecting a request inside T_u to one outside, we have

$$\Delta|\operatorname{sur}^*(u) - \operatorname{sur}(u)| \leq |\Delta(\operatorname{sur}^*(u) - \operatorname{sur}(u))| \leq k + 1 \tag{2}$$

[3] If both cases hold, or if one of the cases holds for more than one v', then pick an arbitrary one.

First, consider the case where $\Delta|\operatorname{sur}^*(u)-\operatorname{sur}(u)| = k+1$, and therefore, $\Delta\phi = 4(k+1)\cdot d_u$. Now both inequalities in (2) are tight. Because the second inequality is tight, all the k requests inside T_u which SOL connected outside must be negative, and $\Delta(\operatorname{sur}^*(u)-\operatorname{sur}(u)) = k+1 > 0$. Furthermore, $\operatorname{sur}^*(u) - \operatorname{sur}(u)$ never decreases during the phase. Because the first inequality in (2) is tight, the sign of $\operatorname{sur}^*(u) - \operatorname{sur}(u)$ at the beginning of the phase must be the same as that of $\Delta(\operatorname{sur}^*(u) - \operatorname{sur}(u))$, implying $\operatorname{sur}^*(u) - \operatorname{sur}(u) \geq 0$ initially. Since $\operatorname{sur}^*(u) - \operatorname{sur}(u)$ never decreases, we have $\operatorname{sur}^*(u) - \operatorname{sur}(u) \geq 0$ throughout the phase. Therefore, at any moment when z_u^+ was increasing, we have $\operatorname{sur}^*(u) \geq \operatorname{sur}(u) > 0$. Thus, the rate of increase of x_u is always at least as much as the rate of increase of z_u^+. Since z_u^+ increases by $2d_u$, we have $\Delta x_u \geq 2d_u$. Therefore,

$$\Delta(y_u^+ + y_u^-) + \Delta\phi \leq 4d_u + 4(k+1)\cdot d_u = 4(2d_u + kd_u) \leq 4\Delta(x_u + x'_u)$$

Next, suppose that $\Delta|\operatorname{sur}^*(u)-\operatorname{sur}(u)| < k+1$. Observe that the parity of $\operatorname{sur}^*(u)-\operatorname{sur}(u)$ changes $k+1$ times during the phase: each time when the algorithm or SOL connects a request in T_u to one outside. Thus, if $\Delta|\operatorname{sur}^*(u) - \operatorname{sur}(u)|$ is not $k+1$, it must be at most $k-1$, which means $\Delta\phi \leq 4(k-1)\cdot d_u$. Therefore,

$$\Delta(y_u^+ + y_u^-) + \Delta\phi \leq 4d_u + 4(k-1)\cdot d_u = 4kd_u = 4\Delta x'_u \leq 4\Delta(x_u + x'_u)$$

Thus, in any case, (1) holds for any complete phase.

Finally, consider the last incomplete phase, which does not have a usage of e_u by the algorithm at the end. Note that at the end of the algorithm's run, $\operatorname{sur}(u) = \operatorname{sur}^*(u) = 0$, and hence, $\phi = 0$. Since ϕ is non-negative by definition, we have $\Delta\phi \leq 0$. If $k > 0$, then $\Delta(x_u + x'_u) \geq \Delta x'_u = kd_u \geq d_u$. Since $\Delta(y_u^+ + y_u^-) \leq 4d_u$, (1) holds. On the other hand, if $k = 0$, then $\operatorname{sur}^*(u) - \operatorname{sur}(u)$ stays constant in the phase. Since it is zero finally, it is zero throughout the phase. Thus, $\operatorname{sur}^*(u) = \operatorname{sur}(u)$ in the entire phase. Since $y_u^+ + y_u^-$ increases at a rate at most $|\operatorname{sur}(u)|$ and x_u increases at a rate at least $|\operatorname{sur}^*(u)|$, we have $\Delta(y_u^+ + y_u^-) \leq \Delta x_u$, again implying (1). ◀

Global and Fixed-Terminal Cuts in Digraphs[*][†]

Kristóf Bérczi[1], Karthekeyan Chandrasekaran[2], Tamás Király[3], Euiwoong Lee[4], and Chao Xu[5]

1 **MTA-ELTE Egerváry Research Group, Budapest, Hungary**
`berkri@cs.elte.hu`

2 **University of Illinois, Urbana-Champaign, IL, USA**
`karthe@illinois.edu`

3 **MTA-ELTE Egerváry Research Group, Budapest, Hungary**
`tkiraly@cs.elte.hu`

4 **Carnegie Mellon University, Pittsburgh, PA, USA**
`euiwoonl@cs.cmu.edu`

5 **University of Illinois, Urbana-Champaign, IL, USA**
`chaoxu3@illinois.edu`

Abstract

The computational complexity of multicut-like problems may vary significantly depending on whether the terminals are fixed or not. In this work we present a comprehensive study of this phenomenon in two types of cut problems in directed graphs: double cut and bicut.

1. Fixed-terminal edge-weighted double cut is known to be solvable efficiently. We show that fixed-terminal node-weighted double cut cannot be approximated to a factor smaller than 2 under the Unique Games Conjecture (UGC), and we also give a 2-approximation algorithm. For the global version of the problem, we prove an inapproximability bound of 3/2 under UGC.
2. Fixed-terminal edge-weighted bicut is known to have an approximability factor of 2 that is tight under UGC. We show that the global edge-weighted bicut is approximable to a factor strictly better than 2, and that the global node-weighted bicut cannot be approximated to a factor smaller than 3/2 under UGC.
3. In relation to these investigations, we also prove two results on undirected graphs which are of independent interest. First, we show NP-completeness and a tight inapproximability bound of 4/3 for the node-weighted 3-cut problem under UGC. Second, we show that for constant k, there exists an efficient algorithm to solve the minimum $\{s, t\}$-separating k-cut problem.

Our techniques for the algorithms are combinatorial, based on LPs and based on the enumeration of approximate min-cuts. Our hardness results are based on combinatorial reductions and integrality gap instances.

1998 ACM Subject Classification G.2.2 Graph Theory

Keywords and phrases Directed Graphs, Arborescence, Graph Cuts, Hardness of Approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.2

1 Introduction

The minimum two-terminal cut problem (min $s-t$ cut) and its global variant (min cut) are classic interdiction problems with fast algorithms. Generalizations of the fixed-terminal

[*] A full version of the paper is available at `https://arxiv.org/abs/1612.00156`.

[†] Kristóf and Tamás are supported by the Hungarian National Research, Development and Innovation Office – NKFIH grants K109240 and K120254. Chao is supported in part by NSF grant CCF-1526799.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 2; pp. 2:1–2:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

variant, including the multi-cut and the multi-way cut, as well as generalizations of the global variant, including the k-cut, have been well-studied in the algorithmic literature [10, 14]. In this work, we study two generalizations of global cut problems to directed graphs, namely double cut and bicut (that we describe below). We study the power and limitations of fixed terminal variants of these cut problems in order to solve the global variants. In the process, we examine "intermediate" multicut problems where only a subset of the terminals are fixed, and obtain results of independent interest. In particular, we show that the undirected $\{s,t\}$-separating k-cut problem, where two of the k terminals are fixed, is polynomial-time solvable for constant k. In what follows, we describe the problems along with the results. We refer the reader to Tables 1, 2, and 3 at the end of Section 1.1 for a summary of the results. We mention that all our algorithmic/approximation results hold for the min-cost variant while the inapproximability results hold for the min-cardinality variant by standard modification of our reductions and algorithms. For ease of presentation, we do not make this distinction.

The starting point of this work is node-weighted double cut, that we describe below. We recall that an arborescence in a directed graph $D = (V, E)$ is a minimal subset $F \subseteq E$ of arcs such that there exists a node $r \in V$ with every node $u \in V$ having a unique path from r to u in the subgraph (V, F) (e.g., see [26]).

Double Cut. The input to the NodeDoubleCut problem is a directed graph and the goal is to find the smallest number of nodes whose deletion ensures that the remaining graph has no arborescence. NodeDoubleCut is a generalization of node weighted global min cut in undirected graphs to directed graphs. It is non-monotonic under node deletion. This problem is key to understanding fault tolerant consensus in networks. We briefly describe this connection.

Significance of double cut. In a recent work, Tseng and Vaidya [28] showed that *consensus* in a directed graph can be achieved in the *synchronous model* subject to the failure of f nodes *if and only if* the removal of any f nodes still leaves an arborescence in the remaining graph. Thus, the number of nodes whose failure can be tolerated for the purposes of achieving consensus in a network is *exactly* one less than the smallest number of nodes whose removal ensures that there is no arborescence in the network. So, it is imperative for the network authority to be able to compute this number.

A directed graph $D = (V, E)$ has no arborescence if and only if [1] there exist two distinct nodes $s, t \in V$ such that every node $u \in V$ can reach at most one node in $\{s, t\}$. By this characterization, every directed graph that does not contain a tournament has a feasible solution to NodeDoubleCut. This characterization motivates the following fixed-terminal variant, denoted $\{s,t\}$-NodeDoubleCut: Given a directed graph with two specified nodes s and t, find the smallest number of nodes whose deletion ensures that every remaining node u can reach at most one node in $\{s, t\}$ in the resulting graph. An instance of $\{s,t\}$-NodeDoubleCut has a feasible solution provided that the instance has no edge between s and t. An efficient algorithm to solve/approximate $\{s,t\}$-NodeDoubleCut immediately gives an efficient algorithm to solve/approximate NodeDoubleCut.

[1] We believe that this characterization led earlier authors [3] to coin the term *double cut* to refer to the edge deletion variant of the problem. We are following this naming convention.

Edge-weighted case. In the edge-weighted version of the problem, $\{s,t\}$-EdgeDoubleCut, the goal is to delete the smallest number of edges to ensure that every node in the graph can reach at most one node in $\{s,t\}$. Similarly, in the global variant, denoted EdgeDoubleCut, the goal is to delete the smallest number of edges to ensure that there exist nodes s,t such that every node u can reach at most one node in $\{s,t\}$, i.e. the graph has no arborescence. The fixed-terminal variant $\{s,t\}$-EdgeDoubleCut is solvable in polynomial time using maximum flow and, consequently, EdgeDoubleCut is also solvable in polynomial time (see e.g. [3]).

Results for double cut. Our main result on the fixed-terminal variant, namely $\{s,t\}$-NodeDoubleCut, is the following hardness of approximation.

▶ **Theorem 1.** $\{s,t\}$-NodeDoubleCut *is NP-hard, and has no efficient* $(2-\epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

We also give a 2-approximation algorithm for $\{s,t\}$-NodeDoubleCut, which leads to a 2-approximation for the global variant.

▶ **Theorem 2.** *There exists an efficient* 2*-approximation algorithm for* $\{s,t\}$-NodeDoubleCut *and* NodeDoubleCut*.*

While we are aware of simple combinatorial algorithms to achieve the 2-approximation for $\{s,t\}$-NodeDoubleCut, we present an LP-based algorithm since it also helps to illustrate an integrality gap instance which is the main tool underlying the hardness of approximation (Theorem 1) for the problem. Next we focus on the complexity of NodeDoubleCut. We note that the NP-hardness of the fixed-terminal variant does not necessarily mean that the global variant is also NP-hard.

▶ **Theorem 3.** NodeDoubleCut *is NP-hard, and has no efficient* $(3/2-\epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

Bicuts offer an alternative generalization of min cut to directed graphs. The approximability of the fixed-terminal variant of bicut is well-understood while the complexity of the global variant is unknown. In the following we describe these bicut problems and exhibit a dichotomic behaviour between the fixed-terminal and the global variant.

Bicut. The edge-weighted two-terminal bicut, denoted $\{s,t\}$-EdgeBiCut, is the following: Given a directed graph with two specified nodes s and t, find the smallest number of edges whose deletion ensures that s cannot reach t and t cannot reach s in the resulting graph. Clearly, $\{s,t\}$-EdgeBiCut is equivalent to 2-terminal multiway-cut (the goal in k-terminal multiway cut is to delete the smallest number of edges to ensure that the given k terminals cannot reach each other). This problem has a rich history and has seen renewed interest in the last few months culminating in inapproximability results matching the best-known approximability factor: $\{s,t\}$-EdgeBiCut admits a 2-factor approximation (by simple combinatorial techniques) and has no efficient $(2-\epsilon)$-approximation assuming the Unique Games Conjecture [19, 5]. In the global variant, denoted EdgeBiCut, the goal is to find the smallest number of edges whose deletion ensures that there exist two distinct nodes s and t such that s cannot reach t and t cannot reach s in the resulting digraph.

The dichotomy between global cut problems and fixed-terminal cut problems in undirected graphs is well-known. For concreteness, we recall Edge-3-Cut and Edge-3-way-Cut. In Edge-3-Cut, the goal is to find the smallest number of edges to delete so that the resulting graph has at least 3 connected components. In Edge-3-way-Cut, the input is an undirected graph with 3 specified nodes and the goal is to find the smallest number of edges to delete so that the resulting graph has at least 3 connected components with at most one of the 3 specified nodes in each. While Edge-3-way-Cut is NP-hard [10], Edge-3-Cut is solvable efficiently [14]. However, such a dichotomy is unknown for directed graphs. In particular, it is unknown whether EdgeBiCut is solvable efficiently. Our next result shows evidence of such a dichotomic behaviour.

Results for bicut. While $\{s, t\}$-EdgeBiCut is inapproximable to a factor better than 2 assuming UGC, we show that EdgeBiCut is approximable to a factor strictly better than 2.

▶ **Theorem 4.** *There exists an efficient* $(2 - 1/448)$*-approximation algorithm for* EdgeBiCut.

We also consider the node-weighted variant of bicut, denoted NodeBiCut: Given a directed graph, find the smallest number of nodes whose deletion ensures that there exist nodes s and t such that s cannot reach t and t cannot reach s in the resulting graph. Every directed graph that does not contain a tournament has a feasible solution to NodeBiCut. NodeBiCut is non-monotonic under node deletion, and it admits a 2-approximation by a simple reduction to $\{s, t\}$-EdgeBiCut. We show the following inapproximability result.

▶ **Theorem 5.** NodeBiCut *is NP-hard, and has no efficient* $(3/2 - \epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

We observe that our approximability and inapproximability factors for NodeDoubleCut and NodeBiCut coincide – 2 and $(3/2 - \epsilon)$ respectively (Theorems 2, 3 and 5).

1.1 Additional Results on Sub-problems and Variants

In what follows, we describe additional results that concern sub-problems in our algorithms/hardness results, and also variants of these sub-problems which are of independent interest.

Node weighted 3-Cut. We show the NP-hardness of NodeDoubleCut in Theorem 3 by a reduction from the node-weighted 3-cut problem in undirected graphs. In the node weighted 3-cut problem, denoted Node-3-Cut, the input is an undirected graph and the goal is to find the smallest subset of nodes whose deletion leads to at least 3 connected components in the remaining graph. A classic result of Goldschmidt and Hochbaum [14] showed that the edge-weighted variant, denoted Edge-3-Cut (see above for definition) – more commonly known as 3-cut – is solvable in polynomial time. Intriguingly, the complexity of Node-3-Cut remained open until now. We present the first results on the complexity of Node-3-Cut.

▶ **Theorem 6.** Node-3-Cut *is NP-hard, and has no efficient* $(4/3 - \epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

The inapproximability factor of 4/3 mentioned in the above theorem is tight: the 4/3-approximation factor can be achieved by guessing 3 terminals that are separated and then using well-known approximation algorithms to solve the resulting node-weighted 3-terminal cut instance [13].

$(s, *, t)$-EdgeLin3Cut. As a sub-problem in the algorithm for Theorem 4, we consider the following, denoted $(s, *, t)$-EdgeLin3Cut (abbreviating edge-weighted linear 3-cut): Given a directed graph $D = (V, E)$ and two specified nodes $s, t \in V$, find the smallest number of edges to delete so that there exists a node r with the property that s cannot reach r and t, and r cannot reach t in the resulting graph. This problem is a global variant of (s, r, t)-EdgeLin3Cut, introduced in [11], where the input specifies three terminals s, r, t and the goal is to find the smallest number of edges whose removal achieves the property above. A simple reduction from Edge-3-way-Cut shows that (s, r, t)-EdgeLin3Cut is NP-hard. The approximability of (s, r, t)-EdgeLin3Cut was studied by Chekuri and Madan [5]. They showed that the inapproximability factor coincides with the flow-cut gap of an associated *path-blocking linear program* assuming the Unique Games Conjecture.

There exists a simple combinatorial 2-approximation algorithm for (s, r, t)-EdgeLin3Cut. A 2-approximation for $(s, *, t)$-EdgeLin3Cut can be obtained by guessing the terminal r and using the above-mentioned approximation. For our purposes, we need a strictly better than 2-approximation for $(s, *, t)$-EdgeLin3Cut; we obtain the following improved approximation factor.

▶ **Theorem 7.** *There exists an efficient* $3/2$*-approximation algorithm for* $(s, *, t)$-EdgeLin3Cut.

$\{s, t\}$-SepEdge$k$Cut. In contrast to (s, r, t)-EdgeLin3Cut, we do not have a hardness result for $(s, *, t)$-EdgeLin3Cut. Upon encountering cut problems in directed graphs, it is often insightful to consider the complexity of the analogous problem in undirected graphs. Our next result shows that the following analogous problem in undirected graphs is in fact solvable in polynomial time: given an undirected graph with two specified nodes s, t, remove the smallest subset of edges so that the resulting graph has at least 3 connected components with s and t being in different components. More generally, we consider $\{s, t\}$-SepEdge$k$Cut, where the goal is to delete the smallest subset of edges from a given undirected graph so that the resulting graph has at least k connected components with s and t being in different components. The complexity of $\{s, t\}$-SepEdge$k$Cut was posed as an open problem by Queyranne [25]. We show that $\{s, t\}$-SepEdge$k$Cut is solvable in polynomial-time for every constant k.

▶ **Theorem 8.** *For every constant k, there is an efficient algorithm to solve* $\{s, t\}$-SepEdge$k$Cut.

$\{s, *\}$-EdgeBiCut. While Theorem 4 shows that EdgeBiCut is approximable to a factor strictly smaller than 2, we do not have a hardness result. We could prove hardness for the following intermediate problem, denoted $\{s, *\}$-EdgeBiCut: Given a directed graph with a specified node s, find the smallest number of edges to delete so that there exists a node t such that s cannot reach t and t cannot reach s in the resulting graph. $\{s, *\}$-EdgeBiCut admits a 2-approximation by guessing the terminal t and then using the 2-approximation for $\{s, t\}$-EdgeBiCut. We show the following inapproximability result:

▶ **Theorem 9.** $\{s, *\}$-EdgeBiCut *is NP-hard, and has no efficient* $(4/3 - \epsilon)$*-approximation for any* $\epsilon > 0$ *assuming the Unique Games Conjecture.*

Due to space constraints, we outline our techniques for the proof of Theorem 4 and for the hardness of approximation results in Sections 2 and 3, and refer the reader to the complete version of this work [2] for all complete proofs. The proofs of Theorems 7 and 8 are presented in Section 4.

Table 1 Global Variants in Directed Graphs. Text in gray refer to known results while text in black refer to the results from this work. All hardness of approximation results are under UGC. Hardness results for Node weighted $(s, *, t)$-LIN3CUT are based on the fact that it is as hard to approximate as Node weighted $\{s, t\}$-SEP3CUT by bidirecting the edges (Table 3).

Problem	Edge-deletion	Node-deletion
DOUBLECUT	Poly-time [3]	2-approx (Thm 2) $(3/2 - \epsilon)$-inapprox (Thm 3)
BICUT	$(2 - 1/448)$-approx (Thm 4)	2-approx $(3/2 - \epsilon)$-inapprox (Thm 5)
$(s, *)$-BICUT	2-approx $(4/3 - \epsilon)$-inapprox (Thm 9)	2-approx $(3/2 - \epsilon)$-inapprox
$(s, *, t)$-LIN3CUT	3/2-approx (Thm 7)	2-approx $(4/3 - \epsilon)$-inapprox

Table 2 Fixed-Terminal Variants in Directed Graphs. Text in gray refer to known results while text in black refer to the results from this work. All hardness of approximation results are under UGC. We include $\{s, t\}$-BICUT and (s, r, t)-LIN3CUT for comparison with the global variants in Table 1.

Problem	Edge-deletion	Node-deletion
(s, t)-DOUBLECUT	Poly-time [3]	2-approx (Thm 2) $(2 - \epsilon)$-inapprox (Thm 1)
(s, t)-BICUT	2-approx $(2 - \epsilon)$-inapprox [4, 19]	[Equivalent to edge-deletion]
(s, r, t)-LIN3CUT	2-approx $(\alpha - \epsilon)$-inapprox [5] (where α is the flow-cut gap)	[Equivalent to edge-deletion]

Table 3 Global Variants in Undirected Graphs. Text in gray refer to known results while text in black refer to the results from this work. All hardness of approximation results are under UGC.

Problem	Edge-deletion	Node-deletion
k-CUT (where k is constant)	Poly-time [14, 18]	$(2 - 2/k)$-approx [13] $(2 - 2/k - \epsilon)$-inapprox (Thm 6)
$\{s, t\}$-SEPkCUT (where k is constant)	Poly-time (Thm 8)	$(2 - 2/k)$-approx [13] $(2 - 2/k - \epsilon)$-inapprox (Thm 6)

1.2 Related Work

In recent work, Bernáth and Pap [3] studied the problem of deleting the smallest number of arcs to block all minimum cost arborescences of a given directed graph. They gave an efficient algorithm to solve this problem through combinatorial techniques. However, their techniques fail to extend to the node weighted double cut problem.

The node-weighted 3-cut problem – NODE-3-CUT – is a generalization of the classic EDGE-3-CUT. Various other generalizations of EDGE-3-CUT have been studied in the literature showing the existence of efficient algorithms. These include the edge-weighted 3-cut in hypergraphs [30, 12] and the more general submodular 3-way partitioning [31, 24]. However, none of these known generalizations address NODE-3-CUT as a special case. Feasible solutions to NODE-3-CUT are also known as shredders in the node-connectivity literature.

In the unit-weight case, shredders whose cardinality is equal to the node connectivity of the graph play a crucial role in the problem of min edge addition to augment node connectivity by one [6, 15, 20, 29]. There are at most linear number of such shredders and all of them can be found efficiently [6, 15]. The complexity of finding a min cardinality shredder was open until our results (Theorem 6).

In the edge-weighted multiway cut in undirected graphs, the input is an undirected graph with k terminal nodes and the goal is to find the smallest cardinality subset of edges whose deletion ensures that there is no path between any pair of terminal nodes. For $k = 3$, a 12/11-approximation is known [7, 16], while for constant k, the current-best approximation factor is 1.2975 due to Sharma and Vondrák [27]. These results are based on an LP-relaxation proposed by Călinescu, Karloff and Rabani [9], known as the CKR relaxation. Manokaran, Naor, Raghavendra and Shwartz [21] showed that the inapproximability factor coincides with the integrality gap of the CKR relaxation. Recently, Angelidakis, Makarychev and Manurangsi [1] exhibited instances with integrality gap at least $6/(5 + (1/k - 1)) - \epsilon$ for every $k \geq 3$ and every $\epsilon > 0$ for the CKR relaxation.

The node-weighted multiway cut in undirected graphs exhibits very different structure in comparison to the edge-weighted multiway cut. It reduces to edge-weighted multiway cut in hypergraphs. Garg, Vazirani and Yannakakis [13] gave a $(2 - 2/k)$-approximation for node-weighted multiway cut by exploiting the extreme point structure of a natural LP-relaxation.

The edge-weighted multiway cut in directed graphs has a 2-approximation, due to Naor and Zosin [23], as well as Chekuri and Madan [4]. Matching inapproximability results were shown recently for $k = 2$ [19, 5]. The node-weighted multiway cut in directed graphs reduces to the edge-weighted multiway cut by exploiting the fact that the terminals are fixed. Such a reduction is unknown for the global version.

1.3 Preliminaries

Let $D = (V, E)$ be a directed graph. For two disjoint sets $X, Y \subset V$, we denote $\delta(X, Y)$ to be the set of edges (u, v) with $u \in X$ and $v \in Y$ and $d(X, Y)$ to be the cut value $|\delta(X, Y)|$. We use $\delta^{in}(X) := \delta(V \setminus X, X)$, $\delta^{out}(X) := \delta(X, V \setminus X)$, $d^{in}(X) := |\delta^{in}(X)|$ and $d^{out}(X) := |\delta^{out}(X)|$. We use a similar notation for undirected graphs by dropping the superscripts. For two nodes $s, t \in V$, a subset $X \subset V$ is called an *$\overline{s}t$-set* if $t \in X \subseteq V - s$. The *cut value* of an $\overline{s}t$-set X is $d^{in}(X)$.

We frequently use the following characterization of directed graphs with no arborescence for the purposes of double cut.

▶ **Theorem 10** (e.g., see [3]). *Let $D = (V, E)$ be a directed graph. The following are equivalent:*

1. *D has no arborescence.*
2. *There exist two distinct nodes $s, t \in V$ such that every node u can reach at most one node in $\{s, t\}$ in D.*
3. *There exist two disjoint non-empty sets $S, T \subset V$ with $\delta^{in}(S) \cup \delta^{in}(T) = \emptyset$.*

2 Overview of approximation for EdgeBiCut

In this section, we present the high-level ideas of the $(2 - 1/448)$-approximation algorithm for EdgeBiCut (Theorem 4). We sketch the argument for a $(2 - \epsilon)$-approximation for some small enough ϵ; the full algorithm and the proof of its approximation ratio are presented in the complete version of this work [2].

Let D be a digraph. For two disjoint sets $X, Y \subset V$, we define $\delta_D(X, Y)$ to be the set of edges (u, v) with $u \in X$ and $v \in Y$ and $d(X, Y)$ to be the cut value $|\delta_D(X, Y)|$. We use $\delta^{in}_D(X) := \delta_D(V \setminus X, X)$, $\delta^{out}_D(X) := \delta_D(X, V \setminus X)$. We drop the subscripts when the graph D is clear from context.

Two sets A and B are called *uncomparable* if $A \setminus B \neq \emptyset$ and $B \setminus A \neq \emptyset$. Given a directed graph $D = (V, E)$, EdgeBiCut is equivalent to finding an uncomparable pair $A, B \subseteq V$ with minimum $|\delta^{in}(A) \cup \delta^{in}(B)|$. Indeed, if A and B are uncomparable and we remove $\delta^{in}(A) \cup \delta^{in}(B)$ from the directed graph, then nodes in $A \setminus B$ cannot reach nodes in $B \setminus A$ and vice versa. On the other hand, if s cannot reach t and t cannot reach s, then the set of nodes that can reach s and the set of nodes that can reach t are uncomparable, and have in-degree 0.

▶ **Definition 11.** For $A, B \subseteq V$, let $\beta(A, B) := |\delta^{in}(A) \cup \delta^{in}(B)|$ and let $\sigma(A, B) := |\delta^{in}(A)| + |\delta^{in}(B)|$. Furthermore, let

$$\beta := \min\{\beta(A, B) \mid A \text{ and } B \text{ are uncomparable}\},$$
$$\sigma := \min\{\sigma(A, B) \mid A \text{ and } B \text{ are uncomparable}\}.$$

As σ can be computed efficiently, we immediately have a $(2-\epsilon)$-approximation if $\sigma \leq (2-\epsilon)\beta$. Also, for fixed $Z \subseteq V$, we can efficiently find an uncomparable pair (A, B) satisfying $A \cap B = Z$ that minimizes $\beta(A, B)$ among pairs with this property, because this is an EdgeDoubleCut problem. The same holds when $V \setminus (A \cup B)$ is fixed. In particular, if there is a pair (A, B) that minimizes $\beta(A, B)$ and $|A \cap B| \leq 2$ or $|V \setminus (A \cup B)| \leq 2$, then we can find the minimizer efficiently. Therefore we assume that every minimizer (A, B) for $\beta(A, B)$ satisfies $|A \cap B| \geq 3$ and $|V \setminus (A \cup B)| \geq 3$. Let us fix one such minimizer (A, B).

In the algorithm, we guess nodes $x \in A \setminus B$, $y \in B \setminus A$, $w_1, w_2 \in V \setminus (A \cup B)$, and $z_1, z_2 \in A \cap B$ (the reason for guessing *two* nodes in the intersection and in the complement of the union is highly technical, and not relevant to this overview). We use the notation $X = A \setminus B$, $Y = B \setminus A$, $W = V \setminus (A \cup B)$, and $Z = A \cap B$.

The algorithm proceeds by making several attempts at finding pairs (A', B') that give a $(2 - \epsilon)$-approximation. Each unsuccessful attempt implies some structural property of the minimum bicut. For example, the first candidate is (X', Y'), where X' is the sink-side of the minimum $\{w_1, w_2, y\} \to \{x, z_1, z_2\}$-cut, and Y' is the sink-side of the minimum $\{w_1, w_2, x\} \to \{y, z_1, z_2\}$-cut. Notice that $\sigma(X', Y') \leq \sigma(A, B)$. If the attempt is unsuccessful, i.e. $\beta(X', Y') > (2 - \epsilon)\beta(A, B)$, then $d(W, Z) > (1 - \epsilon)\beta(A, B) = (1 - \epsilon)\beta$.

Our subsequent attempts are more complex. In our next attempt, we try to expand X' and Y' by the same node set Z' to find $(A' = X' \cup Z', B' = Y' \cup Z')$. Also, we prefer not to have many edges of $E[X'] \cup E[Y']$ in the new bicut (A', B'), because they enter only one among the two sets A' and B', so we make these edges more expensive by duplicating them. Let D_1 be the digraph obtained by duplicating the edges in $E[X'] \cup E[Y']$, and let Z' be the sink-side of the minimum $\{w_1, w_2, x, y\} \to \{z_1, z_2\}$-cut in D_1. It can be shown that the pair $(X' \cup Z', Y' \cup Z')$ is a $(2 - \epsilon)$-approximation unless $|\delta^{in}_{D_1}(Z)| > (2 - 3\epsilon)\beta$.

An analogous attempt can be made by shrinking instead of expanding. Let D_2 be the digraph obtained by duplicating the edges in $E[V \setminus X'] \cup E[V \setminus Y']$, and let W' be the source-side of the minimum $\{w_1, w_2\} \to \{x, y, z_1, z_2\}$-cut in D_2. We obtain that the pair $(X' \setminus W', Y' \setminus W')$ is a $(2 - \epsilon)$-approximation unless $|\delta^{out}_{D_2}(W)| > (2 - 3\epsilon)\beta$.

If the attempts so far are unsuccessful, then $|\delta^{in}_{D_1}(Z)| > (2-3\epsilon)\beta$ and $|\delta^{out}_{D_2}(W)| > (2-3\epsilon)\beta$. From these, it can be shown that all but $O(\epsilon\beta)$ edges in $\delta^{in}(X') \cup \delta^{in}(Y') \cup \delta^{out}(W) \cup \delta^{in}(Z)$ are as positioned in Figure 1.

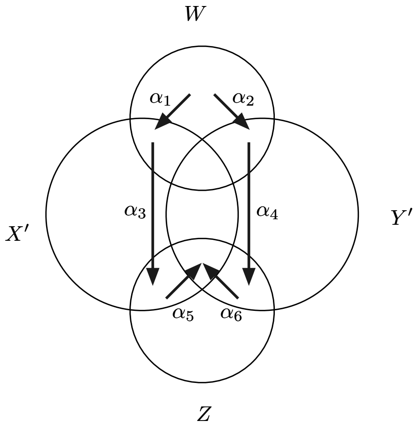


Figure 1 The quantities $\alpha_1, \ldots, \alpha_6$.

Let $\alpha_1, \ldots, \alpha_6$ be the number of edges in each position indicated in Figure 1. We can further show that the quantities $\alpha_1, \alpha_3, \alpha_5$ are within $O(\epsilon\beta)$ of each other, and so are $\alpha_2, \alpha_4, \alpha_6$. Furthermore, $(1 - O(\epsilon))\beta \leq \alpha_3 + \alpha_4 \leq (1 + O(\epsilon))\beta$. W.l.o.g. we may assume $\alpha_3 \geq \alpha_4$.

Our final attempt at obtaining a good bicut is by adding some nodes in $X' \setminus Y'$ to Y' and removing some other nodes of $X' \setminus Y'$ from X'. In other words, our candidate is a pair $(B', Y' \cup A')$ for some $X' \cap Y' \subseteq A' \subsetneq B' \subseteq X'$ (we need the condition $A' \subsetneq B'$ because B' and $Y' \cup A'$ should be uncomparable). When choosing A' and B', we ignore the edges whose contribution does not depend on A' and B'. Let H be the digraph obtained by removing the edges in $E[Y' \cup (V \setminus X')]$. Our aim is to minimize $|\delta_H^{in}(B') \cup \delta_H^{in}(Y' \cup A')|$. However, this quantity differs by $O(\epsilon\beta)$ from $|\delta_H^{in}(A') \cup \delta_H^{in}(B')|$, so we may instead aim to minimize the latter.

The crucial observation is that this minimization problem is an instance of $(s, *, t)$-EdgeLin3Cut. While we do not know how to solve $(s, *, t)$-EdgeLin3Cut optimally, we can efficiently obtain a 3/2-approximation by Theorem 7. By the reformulation of $(s, *, t)$-EdgeLin3Cut in Lemma 13, we get a pair of subsets (A', B') for which $X' \cap Y' \subseteq A' \subsetneq B' \subseteq X'$ and which is a 3/2-approximation. In particular, $|\delta_H^{in}(A') \cup \delta_H^{in}(B')| \leq (3/2)|\delta_H^{in}((X' \cap (Z \cup Y')) \cup \delta_H^{in}(X' \setminus (W \setminus Y'))| \leq 3(\alpha_3 + O(\epsilon)\beta)/2$. Using this and the relationship between the α_i values, we can derive $\beta(B', Y' \cup A') \leq (7/4 + O(\epsilon))\beta$, concluding the proof.

3 Overview of the results on hardness of approximation

Our hardness results include Theorem 1 for $\{s, t\}$-NodeDoubleCut, Theorem 3 for NodeDoubleCut, Theorem 5 for NodeBiCut, Theorem 6 for Node-3-Cut, and Theorem 9 for $\{s, *\}$-EdgeBiCut. We obtain all of our NP-hardness results by reducing from VertexCover on k-regular Graphs, where the input is an undirected k-regular graph, and the goal is to find the smallest subset S of nodes such that every edge in the graph has at least one end-vertex in S. It is APX-hard even for $k = 3$ [8].

We use VertexCover on k-partite Graphs as an intermediate problem, where the input is an undirected k-partite graph $G = (V_1 \cup \cdots \cup V_k, E)$ (we emphasize that the partitioning $V_1, \ldots, V_k$ is specified explicitly in the input) and the goal is to find the smallest subset $S \subset V_1 \cup \cdots \cup V_k$ such that every edge in E has at least one end-vertex in S. Our hardness results are structured as follows.

1. We first show approximation-preserving (combinatorial) reductions from VertexCover on k-regular Graphs (for $k = 3$ or 4) to the above-mentioned problems. These reductions prove all the NP-hardness results. Note that we also get an inapproximability factor of 100/99 and 53/52 respectively under the assumption that $P \neq NP$.
2. For improved hardness of approximation results, we show that VertexCover on k-partite Graphs is hard to approximate within a factor of $2 - 2/k - \epsilon$ for any $\epsilon > 0$ assuming the Unique Games Conjecture. Considering $k = 3$ and $k = 4$, this result in conjunction with the combinatorial reductions show $(4/3-\epsilon)$-inapproximability for NodeDoubleCut and $\{s, *\}$-EdgeBiCut, and $(3/2 - \epsilon)$-inapproximability for NodeBiCut assuming the Unique Games Conjecture.
3. We further improve the hardness of approximation for NodeDoubleCut and $\{s, t\}$-NodeDoubleCut by directly reducing from UniqueGames via the *length-control dictatorship tests* introduced in [19]. We obtain $(3/2 - \epsilon)$-inapproximability for NodeDoubleCut and $(2 - \epsilon)$-inapproximability for $\{s, t\}$-NodeDoubleCut.

In the following section, we sketch the ideas behind the hardness result for $\{s, t\}$-NodeDoubleCut assuming the Unique Games Conjecture.

3.1 $(2 - \epsilon)$-Inapproximability for $\{s, t\}$-NodeDoubleCut

Our results for $\{s, t\}$-NodeDoubleCut and NodeDoubleCut are based on *length-control dictatorship tests* introduced by Lee [19]. Length-control dictatorship tests provide a systematic way to convert integrality gap instances for a natural LP relaxation to dictatorship tests that can be used to prove matching hardness of approximation under the Unique Games Conjecture. In this section we illustrate the high-level ideas behind this conversion for $\{s, t\}$-NodeDoubleCut.

Consider the integrality gap instance $D_{a,b} = (V_D, A_D)$ introduced in Section A (Lemma 20) which shows that the integrality gap of a natural Path-Blocking-LP for $\{s, t\}$-NodeDoubleCut is $2 - o(1)$. We note that $V_D = \{s, t\} \cup ([a] \times [b])$. Let $r = b - 2a + 1$, and $I_D = ([a] \times [b])$ be the set of internal vertices. Furthermore, a good fractional feasible solution as obtained in the proof of Lemma 20 sets $d_v := 1/r$ for every internal vertex v while every integral feasible solution has at least $2a - 1$ vertices in it.

Based on $D_{a,b}$, we define the *dictatorship test* graph $\mathcal{D}^{\mathsf{st}}_{a,b,R,\epsilon} = (V, A)$ as follows, for a positive integer R and $\epsilon > 0$. Consider the probability space (Ω, μ) where $\Omega := \{0, \dots, r-1, *\}$, and $\mu : \Omega \to [0, 1]$ with $\mu(*) = \epsilon$ and $\mu(x) = (1 - \epsilon)/r$ for $x \neq *$.

1. We define $V := \{s, t\} \cup \{v^\alpha_x\}_{\alpha \in I_D, x \in \Omega^R}$. Let v^α denote the set of vertices $\{v^\alpha_x\}_{x \in \Omega^R}$. We also call each v^α as a *hypercube*.
2. For $\alpha \in I_D$ and $x \in \Omega^R$, define the weight as $c(v^\alpha_x) = \prod_{i=1}^{R} \mu(x_i)$. We note that the weight of each hypercube is 1, and the sum of weight of all vertices except s and t is ab. Define the weight of the terminals s and t to be infinite.
3. For each arc between s and $\alpha \in I_D$ in A_D, for each $x \in \Omega^R$, add an arc with the same direction between s and v^α_x. Do the same for each arc between t and $\alpha \in I_D$ in A_D.
4. For each arc $(\alpha, \beta) \in A_D$ with $\alpha = (\alpha_1, \alpha_2), \beta = (\beta_1, \beta_2) \in I_D$ and $x, y \in \Omega^R$, we have an arc from v^α_x to v^β_y according to the following rule (note that $\alpha_2 \neq \beta_2$).
   a. $\alpha_2 < \beta_2$: add an arc if for any $1 \leq j \leq R$: $[y_j = (x_j + 1) \mod r]$ or $[y_j = *]$ or $[x_j = *]$. Call them *forward* arcs.
   b. $\alpha_2 > \beta_2$: add an arc if for any $1 \leq j \leq R$: $[y_j = (x_j - 1) \mod r]$ or $[y_j = *]$ or $[x_j = *]$. Call them *backward* arcs.
   c. If $(\alpha, \beta) \in A_D$ is a jumping arc (as defined in Lemma 20), call (v^α_x, v^β_y) also a jumping arc.

We provide some intuitions behind this conversion. First, we replace each internal vertex $v \in I_D$ by a R-dimensional *hypercube* $v^\alpha = \{v_x^\alpha\}_{x \in \Omega^R}$. Intuitively, our dictatorship test $\mathcal{D}^{\mathsf{st}}_{a,b,R,\epsilon} = (V, A)$, as an instance of $\{s,t\}$-NodeDoubleCut, needs to satisfy the following properties:

1. Completeness: there exists an integral solution $C^* \subseteq V$ of low weight that *reveals an influential coordinate* for each hypercube.
2. Soundness: a subset $C \subseteq V$ is an integral solution of low weight only if it *reveals an influential coordinate* for some hypercube.

To formalize the notion of influence, given $C \subseteq V$ and a hypercube v^α for some $\alpha \in I_D$, let $f = f_{C,\alpha} : \Omega^R \to \{0,1\}$ be such that $f(x) = 1$ if and only if $v_x^\alpha \in C$. Then for each $i \in [R]$, the influence of the ith coordinate is defined by

$$\mathsf{Inf}_i[f] := \mathbb{E}_{x_1,\dots,x_{i-1},x_{i+1},\dots,x_R}[\mathsf{Var}_{x_i}[f(x_1,\dots,x_R)]],$$

where $x_1,\dots,x_R$ are independently sampled from (Ω,μ). For example, suppose C and α are such that $f(x) = 1$ if and only if $x_i = 0$ for some $i \in [R]$ (i.e., f only depends on the ith coordinate), then $\mathsf{Inf}_i[f] = \mu(0) - \mu(0)^2 = \Omega(1/r)$ and $\mathsf{Inf}_j[f] = 0$ for all $j \neq i$, so i is the only influential coordinate. In contrast, suppose C and α are such that $f(x) = 1$ if and only if $x_1 + \cdots + x_R \leq K$ for some K (ignoring x_i with $x_i = *$), then f depends on all coordinates equally and $\mathsf{Inf}_i[f] \to 0$ for all i as $R \to \infty$. We say that C reveals an influential coordinate for the hypercube v^α if $\mathsf{Inf}_i[f_{C,\alpha}]$ is *large* for some $i \in [R]$. Since we will eventually take R to be a sufficiently large constant, a *large influence* means that the influence is some positive constant that does not depend on R.

Given these intuitions, the completeness and soundness properties can be formalized and proved as follows.

Completeness. For the completeness requirement, we need to argue that there exists an integral solution $C^* \subseteq V$ of low weight that reveals an influential coordinate for each hypercube. In particular, we show that a set of vertices that correspond to *dictators* behaves the same as the fractional solution that gives $1/r$ to every vertex and moreover has low weight. For any $q \in [R]$, let $V_q := \{v_x^\alpha : \alpha \in I_D, x_q = * \text{ or } 0\}$. We note that the total weight of V_q is

$$ab\left(\epsilon + \frac{1-\epsilon}{r}\right) \leq ab\epsilon + \frac{ab}{b-2a}.$$

▶ **Lemma 12.** *After removing vertices in V_q, no vertex in V can reach both s and t.*

The proof appears in the full version [2]. The basic intuition is that for any arc from v_x^α to v_y^β for some $\alpha = (\alpha_1,\alpha_2), \beta = (\beta_1,\beta_2), x \in \Omega^R, y \in \Omega^R$, we have $x_q = y_q + 1$ if this arc is going forward ($\alpha_2 < \beta_2$), and $x_q = y_q - 1$ if this arc is going backward ($\alpha_2 > \beta_2$). This relies on our construction and the fact that we removed all v_x^α with $x_q = *$ or $x_q = 0$ since they are in V_q. Since $x_q \in \{1,\dots,r-1\}$, it means that for any path p,

$$|(\text{number of forward arcs in } p) - (\text{number of backward arcs in } p)| < r.$$

As a consequence, this integral solution V_q behaves similar to the fractional solution in D where each internal vertex gets $1/r$, and we can conclude that no vertex can reach both s and t.

Soundness. Suppose that we removed some vertices C such that no vertex $w \in V \setminus C$ can reach both s and t. Our soundness property requires that C either reveals an influential coordinate for some hypercube v^α or $c(C) \geq (2a-1)(1-\epsilon)$. Formally, let τ, d be some constants that depend only on ϵ and r (not R). We prove that if C is a feasible integral solution of $\{s,t\}$-NodeDoubleCut, then either $c(C) \geq (2a-1)(1-\epsilon)$, or $\mathsf{Inf}_i^{\leq d}[f_{C,\alpha}] \geq \tau$ for some $\alpha \in I_D$ and $i \in [R]$. For technical reasons, we use *low-degree influence* $\mathsf{Inf}_i^{\leq d}$ instead of Inf_i.

The main component of the proof is that if there is an arc from α to β in the integrality gap instance $D_{a,b}$, and both $f_{C,\alpha}$, $f_{C,\beta}$ do not reveal an influential coordinate, then we can always find an arc from $v^\alpha \setminus C$ to $v^\beta \setminus C$ in $\mathcal{D}^{\mathsf{st}}$ unless C almost completely contains both v^α and v^β (i.e., $c(C \cap v^\alpha) > 1-\epsilon$ and $c(C \cap v^\beta) > 1-\epsilon$). The proof involves interpreting the set of arcs between two hypercubes as a suitably designed correlated probability space, and using the invariance principle by Mossel [22].

Suppose that C does not reveal an influential coordinate for any hypercube v^α. Then the above fact ensures that for a hypercube v^α, unless it is almost completely contained in C (i.e., $c(C \cap v^\alpha) > 1-\epsilon$), it behaves as if no vertices were contained in C. This observation shows that $c(C)$ must be as large as that of an integral solution in the gap instance $D_{a,b}$. Using the fact that any integral solution of $D_{a,b}$ contains at least $2a-1$ vertices, we conclude that $c(C) \geq (2a-1)(1-\epsilon)$.

In summary, in the completeness case, there exists a subset of vertices of total weight at most $ab\epsilon + ab/(b-2a)$, so that after removing the subset, no vertex can reach both s and t. In the soundness case, unless we reveal an influential coordinate or we remove vertices of total weight at least $(2a-1)(1-\epsilon)$, there exists a vertex that can reach both s and t. The gap between the two cases is at least

$$\frac{(2a-1)(1-\epsilon)}{ab\epsilon + ab/(b-2a)},$$

which approaches to 2 as a increases, by setting $b = a^2$ and $\epsilon = 1/a^4$.

4 EdgeLin3Cut problems

Given a directed graph $D = (V, E)$, a feasible solution to (s,r,t)-EdgeLin3Cut in D is a subset F of arcs whose deletion from the graph eliminates all directed $s \to r$, $r \to t$ and $s \to t$ paths. One of our main tools used in the approximation algorithm for EdgeBiCut is a 3/2-approximation algorithm for $(s,*,t)$-EdgeLin3Cut. We present this algorithm now. For two sets $A, B \subseteq V$, let $\beta(A,B) := |\delta^{in}(A) \cup \delta^{in}(B)|$.

Proof of Theorem 7. We first rephrase the problem in a more convenient way.

▶ **Lemma 13.** *$(s,*,t)$-EdgeLin3Cut in a directed graph $D = (V,E)$ is equivalent to*

$$\min\left\{\beta(A,B) : t \in A \subset B \subseteq V - \{s\}\right\}.$$

Proof. Let $F \subseteq E$ be an optimal solution for $(s,*,t)$-EdgeLin3Cut in D and let $(A,B) := \operatorname{argmin}\{\beta(A,B) : t \in A \subset B \subseteq V - s\}$. Fix an arbitrary node $r \in B - A$. Since the deletion of $\delta^{in}(A) \cup \delta^{in}(B)$ results in a graph with no directed path from s to r, from r to t and from s to t, the edge set $\delta^{in}(A) \cup \delta^{in}(B)$ is a feasible solution to (s,r,t)-EdgeLin3Cut in D, thus implying that $|F| \leq \beta(A,B)$.

On the other hand, F is a feasible solution for (s,r,t)-EdgeLin3Cut in D for some $r \in V - \{s,t\}$. Let A be the set of nodes that can reach t in $D - F$, and R be the set of nodes that can reach r in $D - F$. Then, $F \supseteq \delta^{in}(A)$. Moreover, $F \supseteq \delta^{in}(R \cup A)$ since $R \cup A$ has in-degree 0 in $D - F$, and s is not in $R \cup A$ because it cannot reach r and t in $D - F$. Therefore, taking $B = R \cup A$ we get $F \supseteq \delta^{in}(A) \cup \delta^{in}(B)$. ◀

Our algorithm for determining an optimal pair $(A,B) := \text{argmin}\{\beta(A,B) : t \in A \subset B \subseteq V - s\}$ proceeds as follows: We build a chain $\mathcal{C}$ of $\overline{s}t$-sets with the property that, for some value $k \in \mathbb{Z}_+$,

(i) $\mathcal{C}$ contains only cuts of value at most k, and
(ii) every $\overline{s}t$-set of cut value strictly less than k is in $\mathcal{C}$.

We start with k being the minimum $\overline{s}t$-cut value and $\mathcal{C}$ consisting of a single minimum $\overline{s}t$-cut. In a general step, we find two $\overline{s}t$-sets: a minimum $\overline{s}t$-cut Y compatible with the current chain $\mathcal{C}$, i.e. $\mathcal{C} \cup \{Y\}$ forming a chain, and a minimum $\overline{s}t$-cut Z not compatible with the current chain $\mathcal{C}$, i.e. crossing at least one member of $\mathcal{C}$. These two sets can be found in polynomial time. Indeed, let $t \in C_1 \subset \ldots, \subset C_q \subseteq V - s$ denote the members of $\mathcal{C}$. Find a minimum cut Y_i with $C_i \subseteq Y_i \subseteq V \setminus C_{i+1}$ for $i = 1, \ldots, q$, and choose Y to be a minimum one among these cuts. Concerning Z, for each pair x, y of nodes with $y \in C_i \subseteq V - x$ for some $i \in \{1, \ldots, q\}$, find a minimum cut Z_{xy} with $\{t, x\} \subseteq Z_{xy} \subseteq V - \{s, y\}$, and choose Z to be a minimum one among these cuts. If $d^{in}(Y) \le d^{in}(Z)$, then we add Y to $\mathcal{C}$, and set k to $d^{in}(Y)$; otherwise we set k to $d^{in}(Z)$, and stop.

Let $\mathcal{C}$ denote the chain constructed by the algorithm, and let Y be an arbitrary set crossing some of its members.

▶ **Claim 14.** *$d^{in}(Y) \ge d^{in}(C)$ for all $C \in \mathcal{C}$.*

Proof. Suppose indirectly that $d^{in}(Y) < d^{in}(C)$ for some $C \in \mathcal{C}$. Let $\mathcal{C}'$ denote the chain consisting of those members of $\mathcal{C}$ that were added before C. As C is a set of minimum cut value compatible with $\mathcal{C}'$, Y crosses at least one member of $\mathcal{C}'$. Hence, by $d^{in}(Y) < d^{in}(C)$, the algorithm stops before adding C, a contradiction. ◀

The claim implies that $\mathcal{C}$ satisfies (1) and (2) with the k obtained at the end of the algorithm. Indeed, (1) is obvious from the construction, while (2) follows from the claim and the fact that $\mathcal{C}$ contains all cuts of value strictly less than k that are compatible with $\mathcal{C}$.

By the above, the procedure stops with a chain $\mathcal{C}$ containing all $\overline{s}t$-sets of cut value less than k, and an $\overline{s}t$-set Z of cut value exactly k which crosses some member X of $\mathcal{C}$. If the optimum value of our problem is less than k, then both members of the optimal pair (A,B) belong to the chain $\mathcal{C}$, and we can find them by taking the minimum of $\beta(A',B')$ where $A' \subset B'$ with $A', B' \in \mathcal{C}$.

We can thus assume that the optimum is at least k. As $d^{in}(Z) = k$ and $d^{in}(X) \le k$, the submodularity of the in-degree function implies $d^{in}(X \cap Z) + d^{in}(X \cup Z) \le d^{in}(Z) + d^{in}(X) \le 2k$. Hence at least one of $d^{in}(X \cap Z) \le k$ and $d^{in}(X \cup Z) \le k$ holds. As $d(X \setminus Z, X \cap Z) + d(Z \setminus X, X \cap Z) \le d^{in}(X \cap Z)$ and $d(V \setminus (X \cup Z), X \setminus Z) + d(V \setminus (X \cup Z), Z \setminus X) \le d^{in}(X \cup Z)$, at least one of the following four possibilities is true:

1. $d^{in}(X \cap Z) \le k$ and $d(X \setminus Z, X \cap Z) \le \frac{1}{2}k$. Choose $A = X \cap Z$, $B = X$. Then $\beta(A,B) = d(X \setminus Z, X \cap Z) + d^{in}(X) \le \frac{1}{2}k + k = \frac{3}{2}k$.
2. $d^{in}(X \cap Z) \le k$ and $d(Z \setminus X, X \cap Z) \le \frac{1}{2}k$. Choose $A = X \cap Z$, $B = Z$. Then $\beta(A,B) = d(Z \setminus X, X \cap Z) + d^{in}(Z) \le \frac{1}{2}k + k = \frac{3}{2}k$.
3. $d^{in}(X \cup Z) \le k$ and $d(V \setminus (X \cup Z), X \setminus Z) \le \frac{1}{2}k$. Choose $A = Z$, $B = X \cup Z$. Then $\beta(A,B) = d^{in}(Z) + d(V \setminus (X \cup Z), X \setminus Z) \le k + \frac{1}{2}k = \frac{3}{2}k$.
4. $d^{in}(X \cup Z) \le k$ and $d(V \setminus (X \cup Z), Z \setminus X) \le \frac{1}{2}k$. Choose $A = X$, $B = X \cup Z$. Then $\beta(A,B) = d^{in}(X) + d(V \setminus (X \cup Z), Z \setminus X) \le k + \frac{1}{2}k = \frac{3}{2}k$.

Thus a pair (A, B) can be obtained by taking the minimum among the four possibilities above and $\beta(A', B')$ where $A' \subset B'$ with $A', B' \in \mathcal{C}$, concluding the proof of the theorem. ◀

Next, we show that $\{s,t\}$-SepEdge$k$Cut is solvable in polynomial time if k is a fixed constant.

Let $G = (V, E)$ be an undirected graph. Let the minimum size of an $\{s,t\}$-cut in G be denoted by $\lambda_G(s,t)$. For two subsets of nodes X, Y, let $d(X, Y)$ denote the number of edges between X and Y and let $d(X) := d(X, V \setminus X)$. The cut value of a partition $\{V_1, \ldots, V_q\}$ of V is defined to be the total number of crossing edges, that is, $(1/2)\sum_{i=1}^{q} d(V_i)$, and is denoted by $\gamma(V_1, \ldots, V_q)$. Let $\gamma^q(G)$ denote the value of an optimum Edge-q-Cut in G, i.e.,

$$\min\left\{\gamma(V_1, \ldots, V_q) : V_i \neq \emptyset \ \forall\ i \in [q], V_i \cap V_j = \emptyset \ \forall\ i, j \in [q], \cup_{i=1}^{q} V_i = V\right\}.$$

Proof of Theorem 8. Let γ^* denote the optimum value of $\{s,t\}$-SepEdge$k$Cut in $G = (V, E)$ and let H denote the graph obtained from G by adding an edge of infinite capacity between s and t. The algorithm is based on the following observation (we recommend the reader to consider $k = 3$ for ease of understanding):

▶ **Proposition 15.** *Let $\{V_1, \ldots, V_k\}$ be a partition of V corresponding to an optimal solution of $\{s,t\}$-SepEdge$k$Cut, where s is in V_{k-1} and t is in V_k. Then $\gamma(V_1, \ldots, V_{k-2}, V_{k-1} \cup V_k) \leq 2\gamma^{k-1}(H)$.*

Proof. Let $W_1, \ldots, W_{k-1}$ be a minimum $(k-1)$-cut in H. Clearly, s and t are in the same part, so we may assume that they are in W_{k-1}. Let U_1, U_2 be a minimum $\{s,t\}$-cut in $G[W_{k-1}]$. Then $\{W_1, \ldots, W_{k-2}, U_1, U_2\}$ gives an $\{s,t\}$-separating k-cut, showing that

$$\gamma^* \leq \gamma(W_1, \ldots, W_{k-2}, U_1, U_2) = \gamma^{k-1}(H) + \lambda_{G[W_{k-1}]}(s,t). \tag{1}$$

By Menger's theorem, we have $\lambda_G(s,t)$ pairwise edge-disjoint paths $P_1, \ldots, P_{\lambda_G(s,t)}$ between s and t in G. Consider one of these paths, say P_i. If all nodes of P_i are from $V_{k-1} \cup V_k$, then P_i has to use at least one edge from $\delta(V_{k-1}, V_k)$. Otherwise, P_i uses at least two edges from $\delta(V_1 \cup \cdots \cup V_{k-2}) \cup \bigcup_{\substack{i,j \leq k-2 \\ i \neq j}} \delta(V_i, V_j)$. Hence the maximum number of pairwise edge-disjoint paths between s and t is

$$\lambda_G(s,t) \leq d(V_{k-1}, V_k) + \frac{1}{2}\left(d(V_1 \cup \cdots \cup V_{k-2}) + \sum_{\substack{i,j \leq k-2 \\ i \neq j}} d(V_i, V_j)\right).$$

Thus, we have

$$\begin{aligned}
\gamma^* &= d(V_{k-1}, V_k) + d(V_1 \cup \cdots \cup V_{k-2}) + \sum_{\substack{i,j \leq k-2 \\ i \neq j}} d(V_i, V_j) \\
&\geq \lambda_G(s,t) + \frac{1}{2}\left(d(V_1 \cup \cdots \cup V_{k-2}) + \sum_{\substack{i,j \leq k-2 \\ i \neq j}} d(V_i, V_j)\right) \\
&= \lambda_G(s,t) + \frac{1}{2}\gamma(V_1, \ldots, V_{k-2}, V_{k-1} \cup V_k) \\
&\geq \lambda_{G[W_{k-1}]}(s,t) + \frac{1}{2}\gamma(V_1, \ldots, V_{k-2}, V_{k-1} \cup V_k)
\end{aligned}$$

that is,

$$\gamma^* \geq \lambda_{G[W_{k-1}]}(s,t) + \frac{1}{2}\gamma(V_1,\dots,V_{k-2},V_{k-1}\cup V_k). \tag{2}$$

By combining (1) and (2), we get $\gamma(V_1,\dots,V_{k-2},V_{k-1}\cup V_k) \leq 2\gamma^{k-1}(H)$, proving the proposition. ◀

Karger and Stein [18] showed that the number of feasible solutions to EDGE-k-CUT in G with value at most $2\gamma^k(G)$ is $O(n^{4k})$. All these solutions can be enumerated in polynomial-time for fixed k [18, 17]. This observation together with Proposition 15 gives the following algorithm for finding an optimal solution to $\{s,t\}$-SEPEDGEkCUT:

Step 1. Let H be the graph obtained from G by adding an edge of infinite capacity between s and t. In H, enumerate all feasible solutions to EDGE-$(k-1)$-CUT – namely the vertex partitions $\{W_1,\dots,W_{k-1}\}$ – whose cut value $\gamma_H(W_1,\dots,W_{k-1})$ is at most $2\gamma^{k-1}(H)$. Without loss of generality, assume $s,t \in W_{k-1}$.

Step 2. For each feasible solution to EDGE-$(k-1)$-CUT in H listed in Step 1, find a minimum $\{s,t\}$-cut in $G[W_{k-1}]$, say U_1, U_2.

Step 3. Among all feasible solutions $\{W_1,\dots,W_{k-1}\}$ to EDGE-$(k-1)$-CUT listed in Step 1 and the corresponding U_1, U_2 found in Step 2, return the k-cut $\{W_1,\dots,W_{k-2},U_1,U_2\}$ with minimum $\gamma(W_1,\dots,W_{k-2},U_1,U_2)$.

The correctness of the algorithm follows from Proposition 15: one of the choices enumerated in Step 1 will correspond to the partition $(V_1,\dots,V_{k-2},V_{k-1}\cup V_k)$, where $(V_1,\dots,V_k)$ is the partition corresponding to the optimal solution. ◀

Acknowledgements. Karthik would like to thank Chandra Chekuri, Neil Olver and Chaitanya Swamy for helpful discussions at various stages of this work.

References

1 H. Angelidakis, Y. Makarychev, and P. Manurangsi. An Improved Integrality Gap for the Călinescu-Karloff-Rabani Relaxation for Multiway Cut. Preprint arXiv:1611.05530, 2016. URL: https://arxiv.org/abs/1611.05530.

2 K. Bérczi, K. Chandrasekaran, T. Király, E. Lee, and C. Xu. Global and fixed-terminal cuts in digraphs. Preprint arXiv:1612.00156, 2017. URL: https://arxiv.org/abs/1612.00156.

3 A. Bernáth and G. Pap. Blocking optimal arborescences. In *Proceedings of the 16th International Conference on Integer Programming and Combinatorial Optimization (IPCO)*, pages 74–85, 2013.

4 C. Chekuri and V. Madan. Simple and fast rounding algorithms for directed and node-weighted multiway cut. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'16, pages 797–807, 2016.

5 C. Chekuri and V. Madan. Approximating multicut and the demand graph. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'17, 2017.

6 J. Cheriyan and R. Thurimella. Fast algorithms for k-shredders and k-node connectivity augmentation. *Journal of Algorithms*, 33(1):15–50, 1999.

7 K. Cheung, W. Cunningham, and L. Tang. Optimal 3-terminal cuts and linear programming. *Mathematical Programming*, 106(1):1–23, 2006.

8 M. Chlebík and J. Chlebíková. Complexity of approximating bounded variants of optimization problems. *Theoretical Computer Science*, 354(3):320–338, 2006.

9 G. Călinescu, H. Karloff, and Y. Rabani. An improved approximation algorithm for multiway cut. *Journal of Computer and System Sciences*, 60(3):564–574, 2000.

10 E. Dahlhaus, D. Johnson, C. Papadimitriou, P. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

11 R. Erbacher, T. Jaeger, N. Talele, and J. Teutsch. Directed multicut with linearly ordered terminals. Preprint arXiv:1407.7498, 2014. URL: `https://arxiv.org/abs/1407.7498`.

12 T. Fukunaga. Computing minimum multiway cuts in hypergraphs. *Discrete Optimization*, 10(4):371–382, 2013.

13 N. Garg, V. Vazirani, and M. Yannakakis. Multiway cuts in node weighted graphs. *Journal of Algorithms*, 50(1):49–61, 2004.

14 O. Goldschmidt and D. Hochbaum. A polynomial algorithm for the k-cut problem for fixed k. *Math. Oper. Res.*, 19(1):24–37, Feb 1994.

15 T. Jordán. On the number of shredders. *Journal of Graph Theory*, 31(3):195–200, 1999.

16 D. Karger, P. Klein, C. Stein, M. Thorup, and N. Young. Rounding algorithms for a geometric embedding of minimum multiway cut. *Mathematics of Operations Research*, 29(3):436–461, 2004.

17 D. Karger and R. Motwani. Derandomization through approximation. In *Proceedings of the 26th annual ACM symposium on Theory of computing*, STOC'94, pages 497–506, 1994.

18 D. Karger and C. Stein. A new approach to the minimum cut problem. *Journal of ACM*, 43(4):601–640, July 1996.

19 E. Lee. Improved Hardness for Cut, Interdiction, and Firefighter Problems. Preprint arXiv:1607.05133, 2016. URL: `https://arxiv.org/abs/1607.05133`.

20 G. Liberman and Z. Nutov. On shredders and vertex connectivity augmentation. *Journal of Discrete Algorithms*, 5(1):91–101, 2007.

21 R. Manokaran, J. Naor, P. Raghavendra, and R. Schwartz. SDP Gaps and UGC Hardness for Multiway Cut, 0-extension, and Metric Labeling. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, STOC'08, pages 11–20, 2008.

22 E. Mossel. Gaussian bounds for noise correlation of functions. *Geometric and Functional Analysis*, 19(6):1713–1756, 2010.

23 J. Naor and L. Zosin. A 2-approximation algorithm for the directed multiway cut problem. *SIAM Journal on Computing*, 31(2):477–482, 2001.

24 K. Okumoto, T. Fukunaga, and H. Nagamochi. Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. *Algorithmica*, 62(3):787–806, 2012.

25 M. Queyranne. On Optimum k-way Partitions with Submodular Costs and Minimum Part-Size Constraints. Talk Slides, 2012. URL: `https://smartech.gatech.edu/bitstream/handle/1853/43309/Queyranne.pdf`.

26 A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency.* Algorithms and Combinatorics. Springer, 2003.

27 A. Sharma and J. Vondrák. Multiway cut, pairwise realizable distributions, and descending thresholds. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC'14, pages 724–733, 2014.

28 L. Tseng and N. Vaidya. Fault-Tolerant Consensus in Directed Graphs. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing (PODC 2015)*, pages 451–460, 2015.

29 L. Végh. Augmenting undirected node-connectivity by one. *SIAM J. Discrete Math.*, 25(2):695–718, 2011.

30 M. Xiao. Finding minimum 3-way cuts in hypergraphs. *Information Processing Letters*, 110(14):554–558, 2010.

31 L. Zhao, H. Nagamochi, and T. Ibaraki. Greedy splitting algorithms for approximating multiway partition problems. *Mathematical Programming*, 102(1):167–183, 2005.

A Approximation for NodeDoubleCut

In this section, we present an efficient 2-approximation algorithm for $\{s,t\}$-NodeDoubleCut which also leads to a 2-approximation for NodeDoubleCut by guessing the pair of nodes s, t.

Remark. Our algorithm is LP-based. Although, alternative combinatorial algorithms can be designed for this problem, we provide an LP-based algorithm since it also helps to illustrate an integrality gap instance which is the main tool underlying the hardness of approximation for the problem. Furthermore, it is also easy to round an *optimum* solution to our LP to obtain a solution whose cost is at most twice the *optimum* LP-cost (using complementary slackness conditions). Here, we present a rounding algorithm which starts from *any feasible solution* to the LP (not necessarily optimal) and gives a solution whose cost is at most twice the LP-cost of *that feasible solution*.

At the end of this section, we give an example showing that the integrality gap of the LP nearly matches the approximation factor achieved by our rounding algorithm.

Proof of Theorem 2. We recall the problem: Given a directed graph $D = (V, E)$ with two specified nodes $s, t \in V$ and node costs $c : V \setminus \{s,t\} \to \mathbb{R}_+$, the goal is to find a least cost subset $U \subseteq V \setminus \{s,t\}$ of nodes such that every node $u \in V \setminus U$ can reach at most one node in $\{s,t\}$ in the subgraph $D - U$. We will denote a path P by the set of nodes in the path and the collection of paths from node u to node v by $\mathcal{P}^{u \to v}$. For a fixed function $d : V \to \mathbb{R}_+$, the d-distance of a path P is defined to be $\sum_{u \in P} d_u$ and the shortest d-distance from node u to node v is the minimum d-distance among all paths from node u to node v. We use the following LP-relaxation, where we have a variable d_u for every node $u \in V$:

$$\min \sum_{v \in V \setminus \{s,t\}} c_v d_v \qquad \text{(Path-Blocking-LP)}$$

$$\sum_{v \in P} d_v + \sum_{v \in Q} d_v - d_u \geq 1 \ \forall\ P \in \mathcal{P}^{u \to s}, Q \in \mathcal{P}^{u \to t},\ \forall\ u \in V$$

$$d_s, d_t = 0$$

$$d_v \geq 0 \ \forall\ v \in V$$

We first observe that Path-Blocking-LP can be solved efficiently. The separation problem is the following: given $d : V \to \mathbb{R}_+$, verify if there exists a node $u \in V$ such that the sum of the shortest d-distance path from u to s and the shortest d-distance path from u to t is at most $1 + d_u$. Thus, the separation problem can be solved efficiently by solving the shortest path problem in directed graphs.

Let $d : V \to R_+$ be a feasible solution to Path-Blocking-LP. We now present a rounding algorithm that achieves a 2-factor approximation. We note that our algorithm rounds an arbitrary feasible solution d to obtain an integral solution whose cost is at most twice the LP-cost of the solution d. For a subset U of nodes, let $\Delta^{in}(U)$ be the set of nodes $v \in V \setminus U$ that have an edge to a node $u \in U$.

The rounding algorithm in Figure 2 can be implemented to run in polynomial-time. We first show the feasibility of the solution returned by the rounding algorithm. We use the following claim.

▶ **Claim 16.** *For every $\theta \in (0, 1/2)$, we have $\mathbb{B}^{in}(s,\theta) \cap \mathbb{B}^{in}(t,\theta) = \emptyset$.*

Rounding Algorithm for $\{s, t\}$-NodeDoubleCut

1. Pick θ uniformly from the interval $(0, 1/2)$.
2. Let $\mathbb{B}^{in}(s, \theta)$ and $\mathbb{B}^{in}(t, \theta)$ be the set of nodes whose shortest d-distance to s and t respectively, is at most θ.
3. Return $U := \Delta^{in}(\mathbb{B}^{in}(s, \theta)) \cup \Delta^{in}(\mathbb{B}^{in}(t, \theta))$.

Figure 2 The rounding algorithm for $\{s, t\}$-NodeDoubleCut.

Proof. Say $u \in \mathbb{B}^{in}(s, \theta) \cap \mathbb{B}^{in}(t, \theta)$. Then there exists a path $P \in \mathcal{P}^{u \to s}$ and a path $Q \in \mathcal{P}^{u \to t}$ such that $\sum_{v \in P} d_v + \sum_{v \in Q} d_v \leq 2\theta < 1$, a contradiction to the fact that d is feasible for Path-Blocking-LP. ◀

▶ **Claim 17.** *The solution U returned by the algorithm is such that every node $u \in V \setminus U$ can reach at most one node in $\{s, t\}$ in the subgraph $D - U$.*

Proof. Suppose not. Then there exists $u \in V \setminus U$ that can reach both s and t in $D - U$. If $u \notin \mathbb{B}^{in}(s, \theta)$, then u cannot reach s in $D - U$ since $\mathbb{B}^{in}(s, \theta)$ has no entering edges in $D - U$. Thus, $u \in \mathbb{B}^{in}(s, \theta)$. Similarly, $u \in \mathbb{B}^{in}(t, \theta)$. However, this contradicts the above claim that $\mathbb{B}^{in}(s, \theta) \cap \mathbb{B}^{in}(t, \theta) = \emptyset$. ◀

We next bound the expected cost of the solution returned by the rounding algorithm. Let $\bar{d}(v, a)$ denote the shortest d-distance from node v to node a in D. We use the following claim.

▶ **Claim 18.** *Let $\theta \in (0, 1/2)$. If $v \in \Delta^{in}(\mathbb{B}^{in}(s, \theta))$ then $\theta < \bar{d}(v, s) \leq \theta + d_v$ and $d_v \neq 0$.*

Proof. If $\bar{d}(v, s) \leq \theta$, then $v \in \mathbb{B}^{in}(s, \theta)$, a contradiction to $v \in \Delta^{in}(\mathbb{B}^{in}(s, \theta))$. If $\bar{d}(v, s) > \theta + d_v$, then $v \notin \Delta^{in}(\mathbb{B}^{in}(s, \theta))$, a contradiction. If $d_v = 0$, then $\theta < \bar{d}(v, s) \leq \theta + d_v = \theta$, a contradiction. ◀

▶ **Claim 19.** *For every $v \in V$, the probability that v is chosen in U is at most $2d_v$.*

Proof. The claim holds if $v \in \{s, t\}$. Let us fix $v \in V \setminus \{s, t\}$. By the claim above, if $v \in \Delta^{in}(\mathbb{B}^{in}(s, \theta))$ then $\theta < \bar{d}(v, s) \leq \theta + d_v$ and $d_v \neq 0$. Similarly, if $v \in \Delta^{in}(\mathbb{B}^{in}(t, \theta))$, then $\theta < \bar{d}(v, t) \leq \theta + d_v$ and $d_v \neq 0$. Now, the probability that v is in U is at most

$$\mathsf{Pr}\left(\theta \in \left(\bar{d}(v, s) - d_v, \min\{\bar{d}(v, s), 1/2\}\right) \cup \left(\bar{d}(v, t) - d_v, \min\{\bar{d}(v, t), 1/2\}\right)\right).$$

Without loss of generality, let $\bar{d}(v, s) \leq \bar{d}(v, t)$. We may assume that $d_v > 0$ and $\bar{d}(v, s) - d_v < 1/2$, since otherwise, the probability that v is in U is 0 and the claim is proved. Now, by the feasibility of the solution d to Path-Blocking-LP, we have that $\bar{d}(v, s) + \bar{d}(v, t) - d_v \geq 1$ and hence $\bar{d}(v, t) \geq 1/2$. Therefore,

$$\begin{aligned}
\mathsf{Pr}(v \in U) &\leq \mathsf{Pr}\left(\theta \in \left(\bar{d}(v, s) - d_v, \min(\bar{d}(v, s), 1/2)\right)\right) + \mathsf{Pr}\left(\theta \in \left(\bar{d}(v, t) - d_v, 1/2\right)\right) \\
&= \frac{1}{(1/2)} \left(1/2 - \bar{d}(v, s) + d_v + 1/2 - \bar{d}(v, t) + d_v\right) \\
&= 2\left(1 - (\bar{d}(v, s) + \bar{d}(v, t) - d_v) + d_v\right) \\
&\leq 2d_v.
\end{aligned}$$

The first equality in the above is because θ is chosen uniformly from the interval $(0, 1/2)$ while the last inequality is because of the feasibility of the solution d to Path-Blocking-LP. ◀

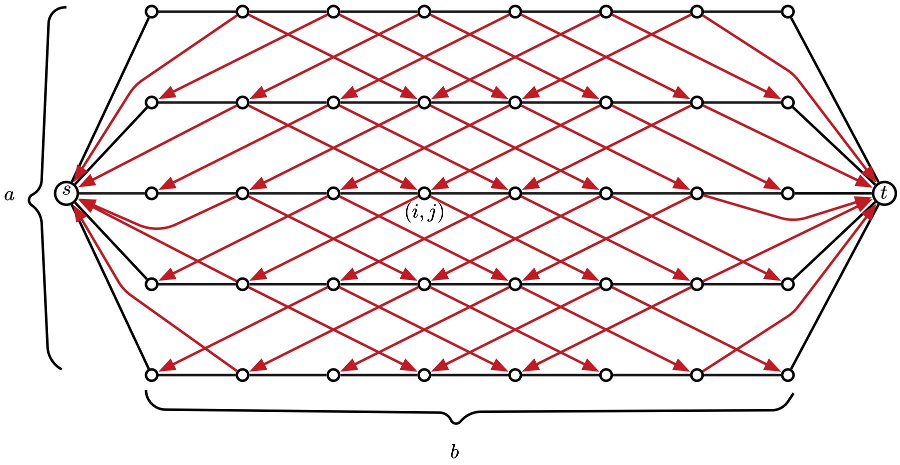


Figure 3 $D_{a,b}$ in the proof of Lemma 20 and $(2-\epsilon)$-inapproximability of $\{s,t\}$-NodeDoubleCut.

By the above claim, the expected cost of the returned solution is

$$\mathbb{E}\left(\sum_{v\in U} c_v\right) = \sum_{v\in V} \mathsf{Pr}(v\in U)c_v \le 2\sum_{v\in V} c_v d_v.$$

Although our rounding algorithm is a randomized algorithm, it can be derandomized using standard techniques. ◀

Our next lemma shows a lower bound on the integrality gap that nearly matches the approximation factor achieved by our rounding algorithm.

▶ **Lemma 20.** *The integrality gap of the Path-Blocking-LP for directed graphs containing n nodes is at least $2 - 7/n^{1/3}$.*

Our integrality gap instance is also helpful in understanding the hardness of approximation of $\{s,t\}$-NodeDoubleCut. So, we define the instance below and summarize its properties which will be used in the proof of Lemma 20 as well as in the proof of hardness of approximation.

For two integers $a, b \in \mathbb{N}$, consider the directed graph $D_{a,b} = (V_D, A_D)$ obtained as follows (see Figure 3): Let $V_D := \{s,t\} \cup ([a] \times [b])$. There are $ab + 2$ nodes. Let $I_D := [a] \times [b]$ and call them as the *internal nodes*. The set of arcs A_D are as follows:

1. For each $1 \le i \le a$, there is a bidirected arc between s and $(i,1)$, and a bidirected arc between (i,b) and t.
2. For each $1 \le i \le a$ and $1 \le j < b$, there is a bidirected arc between (i,j) and $(i,j+1)$.
3. For each $1 \le i < a$ and $2 \le j \le b-1$, there is an arc from (i,j) to $(i+1, j-2)$, and an arc from (i,j) to $(i+1, j+2)$ (let $(i,0) := s$ and $(i, b+1) := t$ for every i). Call them *jumping arcs*.

▶ **Lemma 21.** *$D_{a,b}$ has the following properties:*

1. *For each internal node $\alpha = (\alpha_1, \alpha_2) \in I_D$, each $\alpha \to s$ path has at least $\alpha_2 - a$ internal nodes other than α. Similarly, each $\alpha \to t$ path has at least $b - \alpha_2 - a + 1$ internal nodes other than α.*
2. *If $S \subseteq I_D$ is such that the subgraph induced by $V_D \setminus S$ has no node v that has paths to both s and t, then $|S| \ge 2a - 1$.*

Proof.

1. Jumping arcs are the only arcs that change α_2 by 2 while all other arcs change α_2 by 1. However, a path to s can use at most $a-1$ jumping arcs because they strictly increase α_1. The first property follows from these observations.
2. Suppose that $S \subseteq I_D$ is such that the subgraph induced by $V_D \setminus S$ has no node v that has paths to both s and t. For $i = 1, \ldots, a$, let $s_i := |S \cap \{\{i\} \times [b]\}|$. We note that $s_i \geq 1$ for each i, otherwise s can reach t and t can reach s.
   Suppose $s_i = 1$ for some $1 < i \leq a$ and let j be such that $S \cap \{\{i\} \times [b]\} = (i, j)$. If $j = 1$, then $(i, 2) \in V_D \setminus S$ and $(i, 2)$ can reach both s and t. If $j = b$, then $(i, b-1) \in V_D \setminus S$ and $(i, b-1)$ can reach both s and t. Therefore, we have $1 < j < b$. Then $s_{i-1} \geq 3$ because $(i-1, j-1), (i-1, j), (i-1, j+1)$ can reach both s and t using one jumping arc followed by regular arcs in the ith row.
   Therefore, $|S| = \sum_{i=1}^{a} s_i \geq 1 + 2(a-1) = 2a-1$. ◀

Proof of Lemma 20. The integer optimum of Path-Blocking-LP on $D_{a,b}$ is at least $2a-1$ by the second property of Lemma 21. Let $r := b - 2a + 1$. We set $d_v := 1/r$ for every internal node v. The resulting solution is feasible to Path-Blocking-LP: Indeed, consider $\alpha = (\alpha_1, \alpha_2)$. By the first property of Lemma 21, any $\alpha \to s$ path and $\alpha \to t$ path have to together traverse at least $\alpha_2 - a + (b - \alpha_2 - a + 1) = r$ internal nodes.

Setting $b = a^2$, the integrality gap is at least $(2a-1)/(a^3/r) = 2 - 1/a^3 + 4/a^2 - 5/a \geq 2 - 6/a$ for $a \geq 2$. Using the fact that $a = (|V(D_{a,b})| - 2)^{1/3}$, we get the desired bound on the integrality gap. ◀

A PTAS for Three-Edge-Connected Survivable Network Design in Planar Graphs*†

Glencora Borradaile[1] **and Baigong Zheng**[2]

1 **Oregon State University, Corvallis, OR, USA**
glencora@eecs.oregonstate.edu

2 **Oregon State University, Corvallis, OR, USA**
zhengb@oregonstate.edu

Abstract

We consider the problem of finding the minimum-weight subgraph that satisfies given connectivity requirements. Specifically, given a requirement $r \in \{0, 1, 2, 3\}$ for every vertex, we seek the minimum-weight subgraph that contains, for every pair of vertices u and v, at least $\min\{r(v), r(u)\}$ edge-disjoint u-to-v paths. We give a polynomial-time approximation scheme (PTAS) for this problem when the input graph is planar and the subgraph may use multiple copies of any given edge (paying for each edge separately). This generalizes an earlier result for $r \in \{0, 1, 2\}$. In order to achieve this PTAS, we prove some properties of triconnected planar graphs that may be of independent interest.

1998 ACM Subject Classification G.2.2 [Graph Theory] Graph Algorithms

Keywords and phrases Three-Edge Connectivity, Polynomial-Time Approximation Schemes, Planar Graphs

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.3

1 Introduction

The survivable network design problem aims to find a low-weight subgraph that connects a subset of vertices and will remain connected despite edge failures, an important requirement in the field of telecommunications network design. This problem can be formalized as the I-edge connectivity problem for an integer set I as follows: for an edge-weighted graph G with a requirement function on its vertices $r : V(G) \to I$, we say a subgraph H is a feasible solution if for any pair of vertices $u, v \in V(G)$, H contains $\min\{r(u), r(v)\}$ edge-disjoint u-to-v paths; the goal is to find the cheapest such subgraph. In the *relaxed* version of the problem, H may contain multiple (up to $\max I$) copies of G's edges (H is a *multi*-subgraph) in order to achieve the desired connectivity, paying for the copies according to their multiplicity; otherwise we refer to the problem as the *strict* version. Thus $I = \{1\}$ corresponds to the minimum spanning tree problem and $I = \{0, 1\}$ corresponds to the minimum Steiner tree problem. Here our focus is when $\max I \geq 2$.

This problem and variants have a long history. The I-edge connectivity problem, except when $I = \{1\}$ and $I = \{0\}$, is MAX-SNP-hard [13]. There are constant-factor approximation algorithms for the strict $\{k\}$-edge-connectivity problem: for $k = 2$, Frederickson and Jájá [16] gave a 3-approximation for this problem, and Sebő and Vygen [24] gave a

[*] The full version of this work can be found in [11], http://arxiv.org/abs/1611.03889.

[†] This material is based upon work supported by the National Science Foundation under Grant No. CCF-1252833.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 3; pp. 3:1–3:13
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

4/3-approximation for this problem in unweighted graphs; for any k, Khuller and Vishkin [19] gave a 2-approximation for this problem. Klein and Ravi [23] gave a 2-approximation for the strict $\{0, 1, 2\}$-edge-connectivity problem. For general requirements, Jain [18] gave a 2-approximation for both the strict and relaxed versions of the problem.

We study this problem in planar graphs. In planar graphs, the I-edge connectivity problem, except when $I = \{1\}$ and $I = \{0\}$, is NP-hard (by reduction from Hamiltonian cycle). Berger, Czumaj, Grigni, and Zhao [4] gave a polynomial-time approximation scheme[1] (PTAS) for the relaxed $\{1, 2\}$-edge-connectivity problem, and Berger and Grigni [5] gave a PTAS for the strict $\{2\}$-edge-connectivity problem. Zheng [26] gave a linear PTAS for the strict $\{3\}$-edge-connectivity problem in unweighted planar graphs. Borradaile and Klein [8] gave an *efficient*[2] PTAS (EPTAS) for the relaxed $\{0, 1, 2\}$-edge-connectivity problem[3]. The only planar-specific algorithm for non-spanning, *strict* edge-connectivity is a PTAS for the following problem: given a subset R of edges, find a minimum weight subset S of edges, such that for every edge in R, its endpoints are two-edge-connected in $R \cup S$ [22]; otherwise, the best known results for the strict versions of the edge-connectivity problem when I contains 0 and 2 are the constant-factor approximations known for general graphs.

In this paper, we give an EPTAS for the relaxed $\{0, 1, 2, 3\}$-edge-connectivity problem in planar graphs. This is the first PTAS for connectivity beyond 2-connectivity in planar graphs:

▶ **Theorem 1.** *For any $\epsilon > 0$ and any planar graph instance of the relaxed $\{0, 1, 2, 3\}$-edge connectivity problem, there is an $O(n \log n)$-time algorithm that finds a solution whose weight is at most $1 + \epsilon$ times the weight of an optimal solution.*

In order to give this EPTAS, we must prove some properties of triconnected (three-vertex connected) planar graphs that may be of independent interest. One simple-to-state corollary of the sequel is:

▶ **Theorem 2.** *In a planar graph that minimally pairwise triconnects a set of terminal vertices, every cycle contains at least two terminals.*

In the remainder of this introduction we overview the PTAS framework for network design problems in planar graphs [9] that we use for the relaxed $\{0, 1, 2, 3\}$-edge connectivity problem. In this overview we highlight the technical challenges that arise from handling 3-edge connectivity. We then overview why we use properties of vertex connectivity to address an edge connectivity problem and state our specific observations about triconnected planar graphs that we require for the PTAS framework to apply. In the remainder, 2-EC refers to "relaxed $\{0, 1, 2\}$-edge-connectivity" and 3-EC refers to "relaxed $\{0, 1, 2, 3\}$-edge-connectivity".

1.1 Overview of the planar PTAS framework

The planar PTAS framework grew out of a PTAS for travelling salesperson problem [21] and has been used to give PTASes for Steiner tree [7, 10], Steiner forest [3] and 2-EC [9] problems. For simplicity of presentation, we follow the PTAS whose running time is doubly

[1] A polynomial-time approximation scheme for an minimization problem is an algorithm that, given a fixed constant $\epsilon > 0$, runs in polynomial time and returns a solution within $1 + \epsilon$ of optimal. The algorithm's running time need not be polynomial in ϵ.

[2] A PTAS is efficient if the running time is bounded by a polynomial whose degree is independent of ϵ.

[3] Note that Borradaile and Klein [8] claimed their PTAS would generalize to relaxed $\{0, 1, \ldots, k\}$-edge-connectivity, but this did not come to fruition.

exponential in $1/\epsilon$ [7]; this can be improved to singly exponential as for Steiner tree [10]. Note that for all these problems (except Steiner forest, which requires a preprocessing step to the framework), the optimal value OPT of the solution is within a constant factor of the optimal value of a Steiner tree on the same terminal set where we refer to vertices with non-zero requirement as *terminals*. In the following, O_ϵ-notation hides factors depending on ϵ.

The PTAS framework

The PTAS framework for a planar connectivity problem in graph G consists of the following steps. We describe the steps in terms of the relaxed I-edge connectivity problem, which, at this high level, are easy to generalize from the application of this framework to Steiner tree [7] and 2-EC [9]:

Step 1: Find the *spanner* subgraph H (described below) having the properties:
- **(S1)** $w(H) = O_\epsilon(\text{OPT})$, and
- **(S2)** H contains a feasible solution to the connectivity problem of value at most $(1+\epsilon)\text{OPT}$.

To find a $(1+O(\epsilon))$-approximate solution in G, it is sufficient to find a $(1+\epsilon)$-approximate nearly-optimal solution in H by (S2).

Step 2: Decompose the spanner into a set of subgraphs, called *slices*, such that:
- **(A1)** each slice has *branchwidth* $O_\epsilon(1)$,
- **(A2)** the boundary of a slice is a set of cycles and every cycle bounds exactly two slices,
- **(A3)** the weight of all boundary edges is at most ϵOPT.

The slice boundaries correspond to every k^{th} breadth-first level in the dual graph; this gives property (A2). By choosing $k = O_\epsilon(1)$, we get property (A1). Property (A3) follows from (S1) for k sufficiently large.

Step 3: Add artificial terminals to slice boundaries and assign connectivity requirements so that:
- **(B1)** for each slice, there is a feasible solution over the original and artificial terminals whose weight is bounded by the weight of the slice boundary plus the weight of the optimal solution in the slice.
- **(B2)** the union of these slice solutions is a feasible solution for the original original.

This can be done by adding a terminal to a boundary cycle if the cycle separates any two original terminals and assigning this terminal a connectivity requirement equal to the maximum connectivity requirement the cycle separates (e.g. 2 if the cycle separates two terminals each having a connectivity requirement of 2); this process and the fact that edge connectivity is transitive guarantees property (B2). Property (B1) is guaranteed by property (A3) as seen by adding $2\max I$ copies of the slices to a solution in H.

Step 4: Solve the problem with respect to original and artificial terminals in each slice.

By property (A1), we can do this by dynamic programming over the branch decomposition.

Step 5: Return the union of the slice solutions.

We apply this PTAS framework to the 3-EC problem. Algorithmically, the modifications needed for 3-EC (as compared to 2-EC or Steiner tree) are limited to Step 4; we can obtain an $O_\epsilon(n)$-time dynamic program for the I-edge connectivity problem on graphs with branchwidth $O_\epsilon(1)$, which is similar to that for the k-vertex-connectivity spanning subgraph problem in Euclidean space given by Czumaj and Lingas in [12, 13]. We will argue that the spanner construction (with larger constants) is the same as used for Steiner tree and 2-EC; this argument is the bulk of the technical challenge of this work. Borradaile, Klein and Mathieu show that Step 1 can be done in $O_\epsilon(n \log n)$ time [10, 9] and Steps 2 and 3 can be done in $O(n)$ time. Therefore, we will achieve an $O_\epsilon(n \log n)$ running time for 3-EC.

Spanners for connectivity problems

The spanner construction for Steiner tree and 2-EC [10] (and, as we will argue, for 3-EC) starts with finding the *mortar graph* MG of the input graph G. The mortar graph is a grid-like subgraph of G that spans all the terminals and has total weight bounded by $O_\epsilon(1)$ times the minimum weight of a Steiner tree spanning all the terminals (i.e. weight $O_\epsilon(\text{OPT})$). To construct the mortar graph, we first find an approximate Steiner tree connnecting all terminals and recursively add some short paths. Each face of MG is bounded by four $(1+\epsilon)$ approximations to short paths; the subgraph of G that is enclosed by a face of MG is called a *brick*.

A *structure theorem* shows that there is a nearly optimal solution for Steiner tree and 2-EC whose intersection with each brick is a set of non-crossing trees with $O_\epsilon(1)$ leaves that are *portals* (a subset of $O_\epsilon(1)$ designated vertices of the boundary of the brick) [9]. Each such tree can be computed efficiently since each is a Steiner tree with vertices on the boundary of a planar graph (a brick) [14].

We compute the *spanner* subgraph H by starting with the mortar graph, assigning $O_\epsilon(1)$ vertices of each brick boundary to be portals and adding to the spanner all Steiner trees for each subset of portals in each brick. Since there are $O_\epsilon(1)$ Steiner trees per brick and each has weight at most the boundary of the brick, the spanner has weight $O_\epsilon(\text{OPT})$. By the structure theorem, it is sufficient to solve the given problem in the spanner.

Extension to the 3-EC problem

To prove that the PTAS framework extends to 3-edge connectivity, we need to show this construction results in a spanner for 3-EC, that is, that H contains a $(1+\epsilon)$-approximate solution to 3-EC. This is the main technical challenge of this work. We will prove:

▶ **Theorem 3** (Structure Theorem for 3-EC). *For any $\epsilon > 0$ and any planar graph instance (G, w, r) of the 3-EC problem, there exists a feasible solution S in the spanner H such that*

- *the weight of S is at most $(1 + c\epsilon)\text{OPT}$ where c is an absolute constant, and*
- *the intersection of S with the interior of any brick is a set of $O_\epsilon(1)$ trees whose leaves are on the boundary of the brick and each tree has $O_\epsilon(1)$ leaves.*

The *interior* of a brick is the set of brick edges that are not on the boundary of the brick (that is, not in MG). We denote the interior of a brick B by $\text{int}(B)$. Consider a brick B of G whose boundary is a face of MG and consider the intersection of OPT with the interior of this brick, $\text{OPT} \cap \text{int}(B)$. To prove the Structure Theorem, we will show that:

(P1) $\text{OPT} \cap \text{int}(B)$ can be partitioned into a set of trees $\mathcal{T}$ whose leaves are on the boundary of B.

(P2) If we replace any tree in $\mathcal{T}$ with another tree spanning the same leaves, the result is a feasible solution.

(P3) There is another set of $O(1)$ trees $\mathcal{T}'$ and a set of brick boundary edges B' that costs at most a $1+\epsilon$ factor more than $\mathcal{T}$, such that each tree of $\mathcal{T}'$ has $O(1)$ leaves and $(\text{OPT} \setminus \mathcal{T}) \cup \mathcal{T}' \cup B'$ is a feasible solution.

Property P1 implies that we can decompose an optimal solution into a set of trees inside of bricks plus some edges of MG. Property P2 shows that we can treat those trees independently

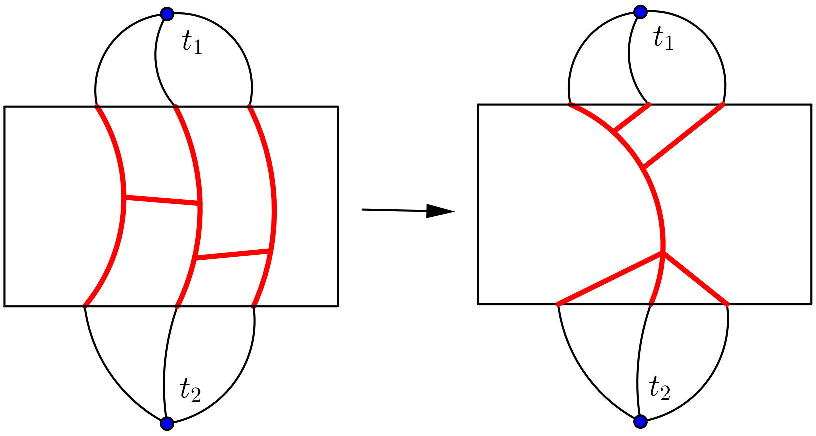


Figure 1 If the bold red tree (left) is $\mathrm{OPT} \cap \mathrm{int}(B)$ (where B is denoted by the rectangle), replacing the tree with another tree spanning the same leaves (right) could destroy 3-connectivity between t_1 and t_2. We will show that such a tree cannot exist in a minimally connected graph.

with regard to connectivity, and this gives us hope that we can replace $\mathrm{OPT} \cap \mathrm{int}(B)$ with some Steiner trees with terminals on the boundary which we can efficiently compute in planar graphs [14]. Property P3 shows that we can compute an approximation to $\mathrm{OPT} \cap \mathrm{int}(B)$ by guessing $O(1)$ leaves.

For the Steiner tree problem, P1 and P2 are nearly trivial to argue; the bulk of the work is in showing P3 [7].

For the 2-EC problem, P1 depends on first converting G and OPT into G' and OPT' such that OPT' biconnects (two-vertex connects) the terminals requiring two-edge connectivity and using the relatively easy-to-argue fact that every cycle of OPT' contains at least one terminal. By this fact, a cycle in OPT' must contain a vertex of the brick's boundary (since MG spans the terminals), allowing the partition of $\mathrm{OPT}' \cap \mathrm{int}(B)$ into trees. P2 and P3 then require that two-connectivity across the brick is maintained.

For the 3-EC problem, P1 is quite involved to show, but further to that, showing Property P2 is also involved; the issues[4] are illustrated in Figure 1 and are the focus of Sections 2 and 3. As with 2-EC, we convert OPT into a vertex connected graph to simplify the arguments. Given Properties P1 and P2, we illustrate Property P3 by following a similar argument as for 2-EC; since this requires reviewing more details of the PTAS framework, we cover this in Section 4.

Non-planar graphs

We point out that, while previously-studied problems that admit PTASes in planar graphs (e.g. independent set and vertex cover [2], TSP [21, 20, 1], Steiner tree [10] and forest [3], 2-EC [9]) generalize to surfaces of bounded genus [6], the method presented in this paper for 3-EC is hard to be generalized to higher genus surfaces. In the generalization to bounded genus surfaces, the graph is preprocessed (by removing some provably unnecessary edges) so that one can compute a mortar graph whose faces bound disks. This guarantees that even though the input graph is not planar, the bricks are; this is sufficient for proving

[4] The issues also appear in 2-ECP, but we explain why it is easy to handle in 2-ECP in the next subsection.

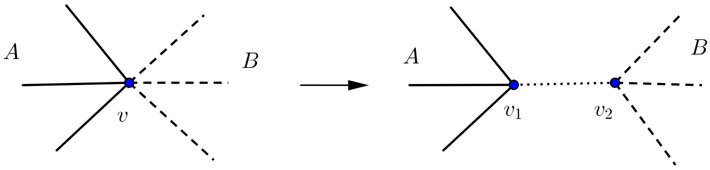


Figure 2 Vertex v is cleaved into vertices v_1 and v_2. The edges incident to v are partitioned into two sets A and B to become incident to distinct copies.

above-numbered properties in the case of TSP, Steiner tree and forest and 2-EC. However, for 3-ECP, in order to prove P2, we require *global* planarity, not just planarity of the brick. To the authors' knowledge, this is the only problem that we know to admit a PTAS in planar graphs that does not naturally generalize to toroidal graphs.

1.2 Reduction to vertex connectivity

Now we overview how we use vertex connectivity to argue about the structural properties of edge-connectivity required for the spanner properties.

We require a few definitions. Vertices x and y are k-vertex-connected in a graph G if G contains k pairwise vertex disjoint x-to-y paths. If $k = 3$ ($k = 2$), then x and y are also called triconnected (biconnected). For a subset Q of vertices in G and a requirement function $r : Q \to \{2, 3\}$, subgraph H is said to be (Q, r)-vertex-connected if every pair of vertices x, y in Q is k-vertex-connected where $k = \min\{r(x), r(y)\}$. We call the vertices of Q *terminals*. If $r(x) = 3$ ($r(x) = 2$) for all $x \in Q$, we say H is Q-triconnected (Q-biconnected). We say a (Q, r)-vertex-connected graph is *minimal*, if no edge or vertex can be deleted without violating the connectivity requirements.

We *cleave* vertices to transform edge-connectivity into vertex-connectivity. Informally, cleaving a vertex is splitting the vertex into two copies and adding a zero-weight edge between the copies; incident edges choose between the copies in a planarity-preserving way (Figure 2). We can cleave the vertices of OPT, creating OPT$'$, so that if two terminals are k-edge-connected in OPT, there are corresponding terminals in OPT$'$ that are k-vertex-connected. We will prove that OPT$'$ satisfies Properties P1 and P2 and since OPT$'$ is obtained from OPT by cleavings, these two properties also hold for OPT.

To prove that OPT$'$ satisfies Property P1, we show that every cycle in OPT$'$ contains at least one terminal (Section 2). To prove that OPT$'$ satisfies Property P2, we define the notion of a *terminal-bounded component*: a connected subgraph is a terminal-bounded component if it is an edge between two terminals or obtained from a maximal terminal-free subgraph S (a subgraph containing no terminals), by adding edges from S to its neighbors (which are all terminals by maximality of S). In Section 3, we show that in a minimal Q-triconnected graph any terminal-bounded component is a tree whose leaves are terminals as well as:

▶ **Theorem 4** (Connectivity Separation Theorem). *Given a minimal (Q, r)-vertex-connected planar graph, for any pair of terminals x and y that require triconnectivity (biconnectivity), there are three (two) vertex disjoint paths from x to y in G such that any two of them do not contain edges of the same terminal-bounded tree.*

▶ **Corollary 5.** *Given a minimal (Q, r)-vertex-connected planar graph, for any pair of terminals x and y that require triconnectivity (biconnectivity), there exist three (two) vertex*

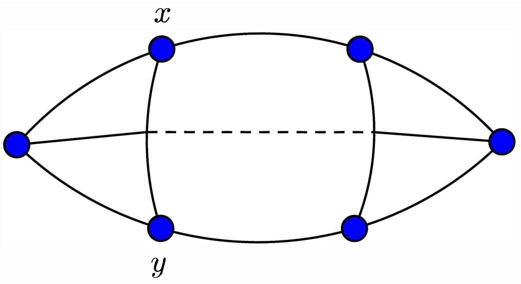


Figure 3 A minimal Q-triconnected graph. The bold vertices are terminals. The dashed path connects two x-to-y paths but it does not contain any terminal.

disjoint x-to-y paths such that any path that connects any two of those x-to-y paths contains a terminal.

This corollary can be viewed as a generalization of the following by Borradaile and Klein for 2-ECP [9]:

▶ **Theorem 6.** (Theorem 2.8 [9]). *Given a graph that minimally biconnects a set of terminals, for any pair of terminals x and y and for any two vertex disjoint x-to-y paths, any path that connects these paths must contain a terminal.*

Note that Theorem 6 holds for general graphs while we only know Corollary 5 to hold for planar graphs, underscoring why our PTAS does not generalize to higher-genus graphs. Further *"for any"* is sufficient for biconnectivity (Theorem 6) whereas *"there exists"* is necessary for triconnectivity (Corollary 5) as illustrated by the example in Figure 3. Higher connectivity comes at a price.

For OPT$'$, Corollary 5 implies Property P2. Consider the set of disjoint paths guaranteed by Corollary 5. If any tree replacement in a brick merges any two disjoint paths, say P_1 and P_2, in the set (the replacement in Figure 1 merges three paths), then the replaced tree must contain at least one vertex of P_1 and one vertex of P_2. This implies the replaced tree contains a P_1-to-P_2 path P such that each vertex in P has degree at least two in the replaced tree. Further, P contains a terminal by Corollary 5. However, all the terminals are in the mortar graph, which forms the boundaries of the bricks. So P must have a common vertex with the boundary of the brick. By Property P1, the replaced tree, which is in the intersection of OPT$'$ with the interior of the brick, can only contain leaves on the boundary of the brick. Therefore, the replaced tree can not contain such a P_1-to-P_2 path, otherwise there is a vertex in P that has degree one in the tree.

2 Vertex-connectivity basics

In this section, we consider minimal (Q, r)-vertex-connected graphs for a subset Q of vertices and a requirement function $r : Q \to \{2, 3\}$.

Borradaile and Klein prove that in a minimal Q-biconnected graph, every cycle contains a terminal (Theorem 2.5 [9]). We show a similar property for a minimal (Q, r)-vertex connected graph here. This property implies property P1, that is the intersection of an optimal solution with the interior of any brick can be partitioned into a set of trees whose leaves are on the boundary of the brick. Note that our proof for this property does not depend on planarity.

For a Q-triconnected graph H, we can obtain another graph H' by contracting all the edges incident to the vertices of degree two in H. We say H' is *contracted version of H* and, alternatively, is *contracted* Q-triconnected. We can prove that H' is triconnected. Further, if H is a minimal Q-triconnected graph, then the contracted version of H is also a minimal Q-triconnected graph. And if $|Q| > 3$, then we can prove H' is simple by the result of Eswaran and Tarjan [15].

Holton, Jackson, Saito and Wormald study the *removability* of edges in triconnected graphs [17]. For an edge $e = uv$ of a simple, triconnected graph G, removing e consists of (i) deleting uv from G, (ii) if u or v now have degree 2, contracting incident edges, and (iii) deleting parallel edges. If the resulting graph after removing e is triconnected, then e is said to be *removable.*

By applying several results of Holton et al. [17] about removable edges, we can prove that every cycle in a minimum contracted Q-triconnected graph contains a terminal. For a graph G that is (Q, r)-vertex connected, let G' be a minimum Q-triconnected graph that is a supergraph of G. Let G'' is the contracted version of G'. Then every cycle in G'' contains a terminal. Since G' is a subdivision of G'', we know every cycle in G' contains a terminal. Since G is a subgraph of G', we have the following theorem.

▶ **Theorem 7.** *For a requirement function $r : Q \to \{2, 3\}$, let G be a minimal (Q, r)-vertex-connected graph. Then every cycle in G contains a vertex of Q.*

3 Connectivity Separation

In this section we continue to focus on vertex connectivity and prove the Connectivity Separation Theorem. The Connectivity Separation Theorem for biconnectivity follows easily from Theorem 6. To see why, consider two paths P_1 and P_2 that witness the biconnectivity of two terminals x and y. For an edge of P_1 to be in the same terminal-bounded component as an edge of P_2, there would need to be a P_1-to-P_2 path that is terminal-free. However, such a path must contain a terminal by Theorem 6. Herein we mainly focus on triconnectivity.

For a requirement function $r : Q \to \{2, 3\}$, let G be a minimal (Q, r)-vertex-connected planar graph. We say a subgraph is *terminal-free* if it is connected and does not contain any terminals. It follows from Theorem 7 that any terminal-free subgraph of G is a tree. We partition the edges of G into *terminal-bounded* components as follows: a terminal-bounded component is either an edge connecting two terminals or is obtained from a *maximal* terminal-free tree T by adding the edges from T to its neighbors, all of which are terminals. Theorem 8 will show that any terminal-bounded subgraph is also a tree.

For a connected subgraph χ of G and an embedding of G with outer face containing no edge of χ, let $C(\chi)$ be the simple cycle that strictly encloses the fewest faces and all edges of χ, if such a cycle exists. (Note that $C(\chi)$ does not exist if there is no aforementioned choice for an outer face.) In order to prove the Connectivity Separation Theorem for bi- and triconnectivity, we start with the following theorem:

▶ **Theorem 8** (Tree Cycle Theorem). *Let T be a terminal-bounded component in a minimal Q-triconnected planar graph H. Then T is a tree and $C(T)$ exists with the following properties*

(a) *The internal vertices of T are strictly inside of $C(T)$.*

(b) *All vertices strictly inside of $C(T)$ are on T.*

(c) *All leaves of T are in $C(T)$.*

(d) *Any pair of distinct maximal terminal-free subpaths of $C(T)$ does not contain vertices of the same terminal-bounded tree.*

Theorem 2 follows from this Tree Cycle Theorem.

Proof of Theorem 2. For a contradiction, assume there is a cycle in H that only containing one terminal, then there is a terminal-bounded component containing that cycle, which can not be a tree, contradicting the Tree Cycle Theorem. ◀

We give an overview of the proof the Tree Cycle Theorem in Subsection 3.2. First, let us see how the Tree Cycle Theorem implies the Connectivity Separation Theorem.

3.1 The Tree Cycle Theorem implies the Connectivity Separation Theorem

For a requirement function $r : Q \to \{2, 3\}$, let G be a minimal (Q, r)-vertex-connected planar graph. Let Q_3 be the set of terminals requiring triconnectivity, and let H be a minimal Q_3-triconnected subgraph of G. Let $Q_2 = Q \setminus Q_3$. Consider two terminals x and y. We sketch the proof here.

Suppose x and y only require biconnectivity. For this case, we know the graph is biconnected and by applying a result of Whitney [25] for the ear decomposition of a biconnected graph, we can find a simple cycle C containing x and y such that every C-to-C path contains a terminal as an internal vertex. As argued at the start of Section 3, this proves the Connectivity Separation Theorem for x and y.

Suppose x and y require triconnectivity, that is $x, y \in Q_3$. Since graph H is Q_3-triconnected, there are three internally vertex-disjoint paths from x to y in H. We modify these three paths such that they do not contain edges of the same terminal-bounded tree. Suppose all three paths contain some edges of a common terminal-bounded tree T. By the Tree Cycle Theorem, there is a cycle $C(T)$ that contains all leaves of T and all other vertices of T are enclosed by $C(T)$. So all the tree paths must intersect cycle $C(T)$. Note that since both of x and y are terminals, the edges incident to x and y are not in the same terminal-bounded tree. So, for each x-to-y path, we can identify non-trivial subpaths: one to-$C(T)$ prefix and one from-$C(T)$ suffix. We can find two subpaths of $C(T)$ and one path in T such that they are vertex-disjoint and the union of these three subpaths together with the to-$C(T)$ prefices and the from-$C(T)$ suffices defines another three internally vertex-disjoint x-to-y paths in H. Only one of the three new paths will contain edges of T. By property (d) of the Tree Cycle Theorem, the two subpaths of $C(T)$ will not introduce any *shared* terminal-bounded tree. We can apply a similar modification when there are only two x-to-y paths containing edges of the same terminal-bounded tree. The argument for extending the property from H to G requires minimal extra work.

3.2 Proof of Tree Cycle Theorem

Let G be a minimal Q_3-triconnected planar graph. We prove the Tree Cycle Theorem for the contracted Q_3-triconnected graph H obtained from G. If the theorem is true for H, then it is true for G since subdivision will maintain the properties of the theorem. We give a high-level overview of the proof.

We focus on a maximal terminal-free tree T^*, rooted arbitrarily, of H and the corresponding terminal-bounded component T (that is, $T^* \subset T$). We show that there is a face of H that does not touch any internal vertex of T^*, which guarantees that there is a drawing of H such that T^* is enclosed by some cycle. We take this face of H as the infinite face. We view T^* as a set $\mathcal{P}$ of root-to-leaf paths. For each path in $\mathcal{P}$, we can find a cycle that strictly encloses only vertices on the paths. The outer cycle of the cycles for all the paths in $\mathcal{P}$ defines $C(T)$. See Figure 4. Property (a) directly follows from the construction. Property (b) is proved

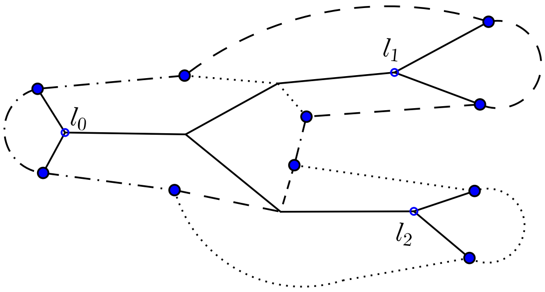


Figure 4 Illustration of $C(T)$. The dashed cycle is C_P for P from l_0 to l_1 and the dotted cycle is $C_{P'}$ for P' from l_0 to l_2. The outer boundary forms $C(T)$.

by induction on the number of root-to-leaf paths of T: when we add a new cycle for a path from $\mathcal{P}$, the new outer cycle will only strictly enclose vertices of the root-to-leaf paths so far considered. After that, we show any two terminals are triconnected when T is a tree: by modifying the three paths between terminals in a similar way to the proof for Connectivity Separation, only one path will require edges in T. Since T is connected, this proves T is a tree by minimality of H. Combining the above properties and triconnectivity of H, we can obtain property (c). Property (d) is proved by contradiction: if there is another terminal-bounded tree T' that shares two terminal-free paths of $C(T)$, then there is a terminal-free path in T'. We can show there is a removable edge in this path of T', contradicting the minimality of H.

4 Proof of the Structure Theorem

In this section, we give a brief overview of the proof of the Structure Theorem (Theorem 3); full details are in the full version of the paper. First we introduce some properties of the mortar graph and bricks. For a brick B, let ∂B be its boundary and $\text{int}(B) = E(B) \setminus E(\partial B)$ be its interior. A path is *ϵ-short* if the distance between every pair of vertices on that path is at most $(1+\epsilon)$ times the distance between them in G. Bricks have the following properties.

▶ **Lemma 9** (Lemma 6.10 [10] rewritten). *The boundary of a brick B, in counterclockwise order, is the concatenation of four paths W_B, S_B, E_B and N_B (west, south, east and north) such that:*

- *Every vertex of $Q \cap B$ is in $N_B \cup S_B$.*
- *N_B is 0-short and every proper subpath of S_B is ϵ-short.*

The paths that form eastern and western boundaries of bricks are called *supercolumns*, and the weight of all edges in supercolumns is at most ϵOPT (Lemma 6.6 [10]). We designate a set of vertices, called *portals*, evenly spaced on the boundary of each brick. Each brick has only constant number (depending on ϵ) of portals on its boundary.

To prove the Structure Theorem, we transform OPT for the instance (G, Q, r) so that it satisfies the following properties (repeated from the introduction):

(P1) $\text{OPT} \cap \text{int}(B)$ can be partitioned into a set of trees $\mathcal{T}$ whose leaves are on the boundary of B.

(P2) If we replace any tree in $\mathcal{T}$ with another tree spanning the same leaves, the result is a feasible solution.

(P3) There is another set of $O(1)$ trees $\mathcal{T}'$ and a set of brick boundary edges B' that costs at most a $1+\epsilon$ factor more than $\mathcal{T}$, such that each tree of $\mathcal{T}'$ has $O(1)$ leaves and $(\text{OPT} \setminus \mathcal{T}) \cup \mathcal{T}' \cup B'$ is a feasible solution.

The transformation consists of the following steps:

Augment. We add four copies of each supercolumn; we take two copies each to be interior to the two adjacent bricks. After this, connectivity between the east and west boundaries of a brick will be transformed to that between the north and south boundaries. Since the weight of all supercolumns is at most ϵOPT, this only increases the weight by an small fraction of OPT.

Cleave. By cleaving a vertex, we split it into multiple copies while keeping the connectivity as required by adding artificial edges of weight zero between two copies and maintaining a planar embedding. We call the resulting solution OPT_C. In this step, we turn k-edge-connectivity into k-vertex-connectivity for $k = 1, 2, 3$. By Theorem 7, we can obtain Property P1: $\text{OPT}_C \cap \text{int}(B)$ can be partitioned into a set $\mathcal{T}$ of trees whose leaves are in ∂B. By Corollary 5, we can obtain Property P2: we can obtain another feasible solution by replacing any tree in $\mathcal{T}$ with another tree spanning the same leaves.

Flatten. For each brick B, we consider the connected components of $\text{OPT}_C \cap \text{int}(B)$. If the component only spans vertices in the north or south boundary, we replace it with the minimum subpath of the boundary that spans the same vertices. This will not increase the weight much by the ϵ-shortness of the north and south boundaries. Note that vertex-connectivity may break as a result, but edge-connectivity is maintained. In the remainder, we only maintain edge-connectivity. We call the resulting solution OPT_F.

Restructure. For each brick B, we consider the connected components of $\text{OPT}_F \cap \text{int}(B)$. We replace each component with a subgraph through a mapping ϕ. The new subgraph may be a tree or a subgraph $\widehat{C}$ that is the union of a cycle and two subpaths of ∂B. The mapping ϕ has the following properties:

- For any component χ of $\text{OPT}_F \cap \text{int}(B)$, $\phi(\chi)$ is connected and spans $\chi \cap \partial B$.
- For two components χ_1 and χ_2 of $\text{OPT}_F \cap \text{int}(B)$, if $\phi(\chi_i) \neq \widehat{C}$ for at least one of $i = 1, 2$, then $\phi(\chi_1)$ and $\phi(\chi_2)$ are edge-disjoint, taking into account edge multiplicities.
- The new subgraph $\phi(\text{OPT}_F \cap \text{int}(B))$ has only constant number (depending on ϵ) of vertices in the boundary ∂B.

We can prove that the total weight is increased by at most ϵOPT_F, giving Property P3. We call the resulting solution OPT_R.

Redirect. We connect each vertex j of $\text{OPT}_R \cap \text{int}(B)$ in the boundary ∂B to the nearest portal p on ∂B by adding multiple copies of the short j-to-p subpath of ∂B. Similar to 2-ECP, we can prove this only increases the weight by an ϵ fraction of OPT and the resulting solution satisfies the Structure Theorem.

Acknowledgements. We thank Hung Le, Amir Nayyeri and David Pritchard for helpful discussions.

References

1 S. Arora, M. Grigni, D. Karger, P. Klein, and A. Woloszyn. A polynomial-time approximation scheme for weighted planar graph TSP. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 33–41, 1998.

2 B. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994. doi:10.1145/174644.174650.

3 M. Bateni, M. Hajiaghayi, and D. Marx. Approximation schemes for Steiner forest on planar graphs and graphs of bounded treewidth. *J. ACM*, 58(5):21, 2011. doi:10.1145/2027216.2027219.

4 A. Berger, A. Czumaj, M. Grigni, and H. Zhao. Approximation schemes for minimum 2-connected spanning subgraphs in weighted planar graphs. In *Proceedings of the 13th European Symposium on Algorithms*, volume 3669 of *Lecture Notes in Computer Science*, pages 472–483, 2005.

5 A. Berger and M. Grigni. Minimum weight 2-edge-connected spanning subgraphs in planar graphs. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming*, volume 4596 of *Lecture Notes in Computer Science*, pages 90–101, 2007. doi:10.1007/978-3-540-73420-8_10.

6 G. Borradaile, E. Demaine, and S. Tazari. Polynomial-time approximation schemes for subset-connectivity problems in bounded-genus graphs. *Algorithmica*, 2012. Online. doi:10.1016/j.jda.2012.04.011.

7 G. Borradaile, C. Kenyon-Mathieu, and P. Klein. A polynomial-time approximation scheme for Steiner tree in planar graphs. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, volume 7, pages 1285–1294, 2007.

8 G. Borradaile and P. Klein. The two-edge connectivity survivable network problem in planar graphs. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*, pages 485–501, 2008.

9 G. Borradaile and P. Klein. The two-edge connectivity survivable-network design problem in planar graphs. *ACM Transactions on Algorithms*, 12(3):30:1–30:29, 2016.

10 G. Borradaile, P. Klein, and C. Mathieu. An $O(n \log n)$ approximation scheme for Steiner tree in planar graphs. *ACM Transactions on Algorithms*, 5(3):1–31, 2009.

11 G. Borradaile and B. Zheng. A PTAS for three-edge-connected survivable network design in planar graphs. *CoRR*, abs/1611.03889, 2016. URL: http://arxiv.org/abs/1611.03889.

12 A. Czumaj and A. Lingas. A polynomial time approximation scheme for euclidean minimum cost k-connectivity. In *Automata, Languages and Programming*, pages 682–694. Springer, 1998.

13 A. Czumaj and A. Lingas. On approximability of the minimum cost k-connected spanning subgraph problem. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 281–290, 1999.

14 R. Erickson, C. Monma, and A. Veinott. Send-and-split method for minimum-concave-cost network flows. *Mathematics of Operations Research*, 12:634–664, 1987.

15 K. Eswaran and R. Tarjan. Augmentation problems. *SIAM Journal on Computing*, 5(4):653–665, 1976.

16 G. Frederickson and J. Jájá. Approximation algorithms for several graph augmentation problems. *SIAM Journal on Computing*, 10(2):270–283, 1981.

17 D. A. Holton, B. Jackson, A. Saito, and N. C. Wormald. Removable edges in 3-connected graphs. *J. Graph Theory*, 14:465–475, 1990.

18 K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 2001(1):39–60, 21.

19 S. Khuller and U. Vishkin. Biconnectivity approximations and graph carvings. *Journal of the ACM*, 41(2):214–235, 1994.

20 P. Klein. A subset spanner for planar graphs, with application to subset TSP. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 749–756, 2006. doi:10.1145/1132516.1132620.

21 P. Klein. A linear-time approximation scheme for TSP in undirected planar graphs with edge-weights. *SIAM Journal on Computing*, 37(6):1926–1952, 2008.

22 P. Klein, C. Mathieu, and H. Zhou. Correlation clustering and two-edge-connected augmentation for planar graphs. In Ernst W. Mayr and Nicolas Ollinger, editors, *32nd International*

Symposium on Theoretical Aspects of Computer Science (STACS 2015), volume 30 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 554–567. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

23 P. Klein and R. Ravi. When cycles collapse: A general approximation technique for constraind two-connectivity problems. In *Proceedings of the 3rd International Conference on Integer Programming and Combinatorial Optimization*, pages 39–55, 1993.

24 A. Sebő and J. Vygen. Shorter tours by nicer ears: 7/5-approximation for the graph-tsp, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. *Combinatorica*, pages 1–34, 2014.

25 H. Whitney. Non-separable and planar graphs. *Trans. Amer. Math. Soc.*, 34:339–362, 1932.

26 B. Zheng. Linear-time approximation schemes for planar minimum three-edge connected and three-vertex connected spanning subgraphs. *CoRR*, abs/1701.08315, 2017. URL: `http://arxiv.org/abs/1701.08315`.

The Quest for Strong Inapproximability Results with Perfect Completeness[*][†]

Joshua Brakensiek[1] and Venkatesan Guruswami[2]

1 **Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA, USA**
`jbrakens@andrew.cmu.edu`

2 **Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA**
`guruswami@cmu.edu`

Abstract

The Unique Games Conjecture (UGC) has pinned down the approximability of all constraint satisfaction problems (CSPs), showing that a natural semidefinite programming relaxation offers the optimal worst-case approximation ratio for any CSP. This elegant picture, however, does not apply for CSP instances that are perfectly satisfiable, due to the imperfect completeness inherent in the UGC. For the important case when the input CSP instance admits a satisfying assignment, it therefore remains wide open to understand how well it can be approximated.

This work is motivated by the pursuit of a better understanding of the inapproximability of perfectly satisfiable instances of CSPs. Our main conceptual contribution is the formulation of a (hypergraph) version of Label Cover which we call "V label cover." Assuming a conjecture concerning the inapproximability of V label cover on perfectly satisfiable instances, we prove the following implications:

- There is an absolute constant c_0 such that for $k \geq 3$, given a satisfiable instance of Boolean k-CSP, it is hard to find an assignment satisfying more than $c_0 k^2/2^k$ fraction of the constraints.
- Given a k-uniform hypergraph, $k \geq 2$, for all $\epsilon > 0$, it is hard to tell if it is q-strongly colorable or has no independent set with an ϵ fraction of vertices, where $q = \lceil k + \sqrt{k} - \frac{1}{2} \rceil$.
- Given a k-uniform hypergraph, $k \geq 3$, for all $\epsilon > 0$, it is hard to tell if it is $(k-1)$-rainbow colorable or has no independent set with an ϵ fraction of vertices.

We further supplement the above results with a proof that an "almost Unique" version of Label Cover can be approximated within a constant factor on satisfiable instances.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases inapproximability, hardness of approximation, dictatorship testing, constraint satisfaction, hypergraph coloring

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.4

1 Introduction

The sustained progress on approximation algorithms and inapproximability results for optimization problems since the early 1990s has been nothing short of extraordinary. This has led to a sharp understanding of the approximability threshold of many fundamental problems, alongside the development of a rich body of techniques on the algorithmic, hardness,

[*] Research supported in part by NSF grants CCF-1422045 and CCF-1526092.
[†] Full version is available at [7], `https://eccc.weizmann.ac.il/report/2017/080/`.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 4; pp. 4:1–4:20
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

and mathematical programming aspects of approximate optimization. Yet there also remain many problems which have resisted resolution and for some there are in fact large gaps between the known algorithmic and hardness results. Examples include vertex cover, graph coloring, max-cut, feedback vertex set, undirected multicut, densest subgraph, and so on.

The Unique Games Conjecture of Khot [38] postulates a strong inapproximability result for a particular class of arity two constraint satisfaction problems. This single assumption has a remarkable array of consequences, and implies *tight* inapproximability results for numerous problems including Vertex Cover [41], max-cut and indeed all constraint satisfaction problems (CSPs) [39, 49, 52], maximum acyclic subgraph and all ordering CSPs [20], scheduling problems [4, 3], graph pricing [44], and cut problems like directed multicut [45], to name a few. Furthermore, for CSPs, the UGC implies that a standard semidefinite programming relaxation gives the best approximation ratio [52, 53, 8].

While the UGC has identified a common barrier against progress on a host of approximation problems, there are still several situations it does not apply to. Crucially, imperfect completeness, where Yes instances are only almost satisfiable, is inherent in the UGC, and this feature is inherited by the problems it reduces to. In particular, the UGC does not say *anything* about problems with *perfect completeness*, where Yes instances have a perfect solution obeying all the constraints. Important classes of such problems include satisfiable instances of CSPs (which have a perfect satisfying assignment and the goal is maximize the number of satisfied constraints) and coloring graphs/hypergraphs with approximately optimal number of colors.

Our understanding of approximating satisfiable instances of CSPs still has many gaps. Håstad's tight hardness result for approximating Max 3-SAT on satisfiable instances was much harder to prove than the analogous result for near-satisfiable instances, and was an early sign of the subtleties of ensuring perfect completeness. The approximability of satisfiable CSPs corresponds via a direct translation to the power of probabilistically checkable proof (PCP) systems with perfect completeness – the best soundness error one can achieve with a k query (non-adaptive) PCP is equal to the best inapproximability factor one can prove for a satisfiable arity k CSP. For $k = 3$, the best soundness is $5/8 + \epsilon$ for any $\epsilon > 0$, and this was established only recently via an intricate proof of the approximation resistance of satisfiable NTW (the arity 3 No-Two predicate which stipulates the number of true literals must be either $0, 1$ or 3) [35]. As a basic open question that still remains wide open, we do not know the approximability of satisfiable Max NAE-3-SAT (not-all-equal 3-SAT) under any plausible (or even not so plausible!) conjecture.

The above-mentioned Unique Games hardness results consist of two components: (i) a dictatorship test that gives a way to test if a function is a dictator or is far from a dictator (e.g., has no influential coordinates), using constraints corresponding to the problem at hand (for NAE-3-SAT this would be checking if certain triples of function values are not all equal), and (ii) a reduction from Unique Games via the dictatorship test that establishes inapproximability under the UGC. The second step is standard, and it gives a "free pass" from the world of combinatorics/analysis of Boolean functions to the complexity world. When we require perfect completeness, no such conjectured off-the-shelf compiler from dictatorship tests to hardness is known (and such a passage even appears unlikely). For instance, dictatorship tests with perfect completeness and optimal soundness are known or Max k-CSP [59] and Max NAE-3-SAT [1]. However, in both cases we do not have matching inapproximability results under any plausible conjecture.

[1] Folklore, and this has connections to robust forms of Arrow's theorem [36] and [50, Sec. 4].

The closest to a UGC surrogate in the literature is the d-to-1 conjecture also made in [38]. The Unique Games problem is an arity 2 CSP whose constraints are bijections; the d-to-1 Label Cover is an arity 2 CSP whose constraints are d-to-1 functions. When $d \geq 2$, deciding satisfiability of a d-to-1 Label Cover instance is NP-complete, unlike Unique Games whose satisfiability is trivial to ascertain. Khot's d-to-1 conjecture states that d-to-1 Label Cover is also hard to approximate within any constant factor, even on satisfiable instances. Note that the UGC and d-to-1 conjecture are incomparable in strength; the UGC has simpler bijective constraints but the d-to-1 conjecture asserts perfect completeness which the UGC cannot.

The d-to-1 conjecture has been used to show some strong inapproximability results with perfect completeness. Such applications are, however, sporadic and also typically do not yield tight results. Some of these results are conditioned specifically on the 2-to-1 conjecture, such as a $\sqrt{2} - \epsilon$ inapproximability for vertex cover (mentioned in [38] and explicit in [40]), Max k-coloring with perfect completeness [27], and coloring 4-colorable graphs [13]. The d-to-1 conjecture, for any fixed d, has been used to show the approximation resistance of NTW [51] and a similar result for larger arity [32],[2] and finding independent sets in 2-colorable 3-uniform hypergraphs [43]. Yet, the implications of the d-to-1 conjecture are limited, and it has become apparent that it is not a versatile starting point for hardness results with perfect completeness.

1.1 Our contributions

Given the above context, our work is motivated by the quest for a better starting point than 2-to-1 Label Cover for inapproximability results with perfect completeness, and which might be able to give striking consequences similar to the UGC.

Aggressive Unique Games variant. One version of Label Cover that is most similar to Unique Games, which we call (L, s)-nearly unique Label Cover, has constraint relations in[3] $[L] \times [L]$ consisting of a matching and s additional edges, for a small s that is a constant independent of L. For this version, it is NP-hard to decide satisfiability, and in fact one can give strong reductions matching the performance of dictatorship tests from it. However, this nearly unique form of Label Cover has a constant factor approximation algorithm with ratio depending only on s. We prove this result in the full version of the paper.

V label cover. Our main conceptual contribution is the formulation of a (hypergraph) version of Label Cover which we call "V label cover." This is an extension of 2-to-1 Label Cover, where the constraint predicates are 2-to-1 maps from $[2L]$ to $[L]$, whose relation graph can be visualized as L disjoint "V's." In V label cover of arity k, we have "longer V's" where the two branches involve k variables which coincide in single variable.[4] This is best illustrated by Figure 1 in Section 3.

We put forth the *V label cover conjecture*, which asserts a strong inapproximability result for this problem. For completeness, we want an assignment where for every constraint, the k variables involved get values in a single "V-branch." For soundness, we insist that no assignment even *weakly satisfies* more than a tiny fraction of constraints, where a constraint

[2] These were later improved to NP-hardness in [35] and [63].

[3] We denote $[L] = \{1, \ldots, L\}$.

[4] We should mention that our path to the formulation of V label cover was more circuitous, and has its origins in attempts to define hypergraph versions of the "α Label Cover" problem of [13].

is weakly satisfied if two of its k variables get values in some V-branch. [5] For this to make sense, the "junction" of the V's cannot all be on the same variable (as in 2-to-1 Label Cover), as in that case we will have a Unique Label Cover constraint between the other $(k-1)$ variables, which we can perfectly satisfy. Therefore, in our V label cover constraints, we have V's with junctions at all the k variables involved in the constraint. At a high level, this is similar to the correlation-breaking constraints of Chan [10].

Near-optimal inapproximability for Max k-CSP with perfect completeness. Assuming the V label cover conjecture, we prove a near-tight inapproximability result for approximating satisfiable Max k-CSP over any fixed domain.

▶ **Theorem 1.1.** *Assume the V label cover conjecture. There is an absolute constant c_0 such that for $k \geq 3$, given a satisfiable instance of Boolean k-CSP, it is hard to find an assignment satisfying more than $c_0 k^2/2^k$ fraction of the constraints. For CSP over domain size $q \geq 3$, where q is a prime power, it is hard to satisfy more than $c_0 k^3 q^3/q^k$ of the constraints.*

The approximability of Max k-CSP has been the subject of many papers in the past two decades since the advent of Håstad's optimal inapproximability results [29]; a partial list includes [58, 57, 16, 30, 17, 56, 25, 2, 10, 33] on the hardness side, and [60, 61, 28, 11, 25, 46] on the algorithmic side.

The best known approximation guarantee for Max k-CSP over domain size q is $\Omega(kq/q^k)$ (for $k \geq \Omega(\log q)$, and $0.62k/2^k$ for the Boolean case [46]. This tight up to constant factors, due to Chan's inapproximability factor of $O(kq/q^k)$ [10]. However, this hardness does not apply for satisfiable instances. For satisfiable instances, the best hardness factor is $2^{O(k^{1/3})}/2^k$ for Boolean Max k-CSP [33], and $q^{O(\sqrt{k})}/q^k$ for Max k-CSP over domain size a prime q [30]. Note that our improved hardness factors (conditioned on the V label cover conjecture) from Theorem 1.1 are the first to get $\mathrm{poly}(k,q)/q^k$ type hardness for satisfiable instances (albeit only for prime powers) and are close to optimal. We note that satisfiable instances can be easier to approximate – Trevisan gave an elegant linear-algebra based factor $(k+1)/2^k$ approximation algorithm for satisfiable Boolean Max k-CSP [61] long before Hast's $\Omega(k/2^k)$ algorithm for the general case [28].

Inapproximability for strong and rainbow colorable hypergraphs. Our other application of the V label cover conjecture is to hypergraph coloring, another fundamental problem where perfect completeness is crucial. We say a hypergraph is c-colorable if there is a coloring of its vertices with c colors so that no hyperedge is monochromatic. Given a 2-colorable k-uniform hypergraph for $k \geq 3$, strong inapproximability results that show the NP-hardness of coloring with any fixed ℓ number of colors are known [21, 14], and recent developments show hardness (for $k \geq 8$) even for $\ell = \exp((\log n)^{\Omega(1)})$ where n is the number of vertices [42, 62, 34]. However, these results do not apply when the hypergraph has some form of balanced coloring that is stronger than just being 2-colorable. Specifically, we consider the notions of strong and rainbow colorability in this work. A hypergraph is q-strongly colorable, $q \geq k$ (resp. q-rainbow colorable, $q \leq k$) if it can be colored with q colors so that in every hyperedge, all vertices get distinct colors (resp. all q colors are represented). We refer the reader to the recent work [23, 6, 5] for further context on these notions. When $k = q$, so that there

[5] This stronger requirement in soundness is common in hypergraph versions of Label Cover. For general Label Cover the stronger soundness guarantee can be ensured with a minor loss in parameters, but for V label cover we do not know such a reduction.

is a perfectly balanced k-coloring where each hyperedge has exactly one vertex of each of the k colors, one can in polynomial time find a 2-coloring without any monochromatic hyperedge [47]. Here we prove a strong hardness result for coloring hypergraphs (in fact for finding sizable independent sets), when this perfect balance condition is relaxed even slightly (specifically, $q = k - 1$ for rainbow coloring, and $q = k + o(k)$ for strong coloring).

A q-strong coloring of a hypergraph is also a legal q-coloring of the graph obtained by converting each of its hyperedges into a clique. For this reason, our hardness result for strongly colorable hypergraphs also implies hardness results in the more elementary setting of *approximate graph coloring.* There are several "pure" NP-hardness results known for graph coloring (e.g., the best known results in different regimes are [37, 22, 34, 6]), but there is a gigantic gap between these results and the known algorithms. [13] establishes much improved results, assuming variants of both the 2–to–1 conjecture as well as a new variant known as *alpha label cover.* Their main result is that for all $\epsilon > 0$, given a 3–colorable graph G, under these assumptions, it is NP–hard to locate an independent set with $|G|\epsilon$ vertices. In this work, assuming the V label cover–conjecture, we give a substantial generalization of this hardness.

▶ **Theorem 1.2.** *Assume the V label cover conjecture.*[6]

- *Given a k-uniform hypergraph, $k \geq 2$, for all $\epsilon > 0$, it is hard to tell if it is q-strongly colorable or has no independent set with an ϵ fraction of vertices, where $q = \lceil k + \sqrt{k} - \frac{1}{2} \rceil$.*
- *Given a k-uniform hypergraph, $k \geq 3$, for all $\epsilon > 0$, it is hard to tell if it is $(k-1)$-rainbow colorable or has no independent set with an ϵ fraction of vertices.*

The authors of [23] showed that for any $\epsilon > 0$, it is NP-hard to distinguish if a k-uniform hypergraph (k even) is a $k/2$-rainbow colorable or does not have a independent set with ϵ fraction of the vertices. The results of [6] give results for strong coloring, but they only apply when $k = 2$ or when the weak coloring has only two colors. Thus, modulo the V label cover–conjecture, our results improve on those in the literature.

These proofs are included in the full version of the paper.

A path to NP-hardness results? In several cases, the UGC conditioned hardness results were later replaced by NP-hardness results. Examples include some geometric inapproximability results [26], hardness of Unique Coverage [24], inapproximability results for agnostic learning [18], tight hardness results for scheduling [55], Chan's breakthrough showing an asymptotically tight inapproximability result for (near-satisfiable) Max k-CSP [10], etc. We hope that establishing a similar body of conditional results for perfect completeness, based on the V label cover conjecture or related variants, will point to strong inapproximability results and spur unconditional results in this domain.

1.2 Proof overview

We now briefly describe the steps needed to prove Theorem 1.1 and Theorem 1.2.

In each case, we reduce from a V label cover instance to a constraint satisfaction problem (with weighted constraints). In Section 3.3, we detail this reduction. The structure of the reduction has the same standard form as many other inapproximability results. Each vertex of the V label cover instance is replaced by a constellation of variables, known as a long code. Each hyperedge of the V label cover instance is replaced by a probability distribution of

[6] Technically, we need an "induced" version of the V label cover conjecture for this result.

constraints between the variables in the correspond long codes. This is done carefully as to ensure that perfectly strongly satisfiable V label cover instances map to perfectly satisfiable CSPs.

For each problem type (Max-k-CSP, strong coloring, rainbow coloring), we craft a probability distribution which exploits its underlying structure. The probability distributions need to have a special correlation structure in order to be compatible with the V label cover constraints. We abstract a general notion termed *V label cover–compatibility* (Definition 3.2) which captures the properties common to these distributions. For example, we dictate that each vertex of each long code is sampled uniformly at random. Then, for each application, we outline the additional properties of our probability distributions in order for the reductions to have the proper soundness (Definition 4.1).

For the soundness analysis, given a good approximation to the resulting CSP, we seek to find an approximate weak labeling of the original V label cover instance. To do that, we attempt to decode each long code by finding one (or many) low-degree influential coordinates; these coordinates can be viewed as candidate labels for the associated vertex. We then argue that for a sizable fraction of constraints, two of the decoded labels will belong to the a single V-branch in the constraint. We can then label our V label cover instance by assigning each vertex a label selected at random from among its decoded labels, which in expectation finds a good approximate weak labeling.

In order to guarantee these influential coordinates, we invoke a couple of invariance principles. For Max-k-CSP, we directly invoke a result due to Mossel (Theorem 2.9) on pairwise independent probability distributions. This version guarantees a common influential coordinate between *three* functions that belongs to a common "V." A pigeonhole principle then implies that two of these labels must be in the same branch. For the hypergraph coloring problems, where we do not have pairwise independence of the distributions, we generalize the invariance principles of Mossel [48, 13] to yield a common influential coordinate for two functions that further lie on the same V-branch. This result, is available in the full version of the paper.

1.3 Organization

In Section 2, we outline the necessary background on CSPs and probability spaces. In Section 3, we motivate and detail the V label cover–conjecture. In Section 4, we apply V label cover to the Max-k-CSP problem. In Appendix A, we prove Lemma 4.4.

2 Preliminaries

2.1 Probability distributions

As is now commonplace in hardness of approximation reductions (e.g., [10, 13, 2, 48]), we utilize the following results on correlated probability spaces.

▶ **Definition 2.1** ([31, 19, 54][7]). Let $X \times Y$ be a finite joint probability space with a probability measure μ. The *correlation* between X and Y, denoted $\rho(X, Y)$ is defined to be

$$\rho(X,Y) = \sup_{\substack{f:X\to\mathbb{R},g:Y\to\mathbb{R}\\ \mathbb{E}[f]=\mathbb{E}[g]=0,\quad \mathrm{Var}[f]=\mathrm{Var}[g]=1}} \left[\mathop{\mathbb{E}}_{(x,y)\sim\mu}[f(x)g(y)]\right].$$

[7] See [1] for a history of this definition.

This is then easily extended to the correlation of $n \geq 3$ spaces.

▶ **Definition 2.2** (Definition 1.9 of [48]). Let $X_1 \times X_2 \times \cdots \times X_n$ be a finite joint probability space. Let $Z_i = X_1 \times X_2 \times \cdots \times X_{i-1} \times X_{i+1} \times \cdots \times X_n$. Then we define the correlation of $X_1, \ldots, X_n$ to be

$$\rho(X_1, X_2, \ldots, X_n) = \max_{1 \leq i \leq n} \rho(X_i, Z_i).$$

When a probability space can be decomposed into the product of independent subspaces, then the correlation behaves elegantly.

▶ **Lemma 2.3** (Theorem 1 of [64]). *For all $i \in [n]$, let $X_i \times Y_i$ be a probability space with measure μ_i. Assume that $\mu_1, \ldots, \mu_n$ are independent. Then,*

$$\rho(X_1 \times X_2 \times \cdots \times X_n, Y_1 \times Y_2 \times \cdots \times Y_n) = \max_{1 \leq i \leq n} \rho(X_i, Y_i).$$

Often it can be difficult to bound the correlation of a distribution away from 1. The following result is key in reducing these complex correlation problems into rather elementary graph connectivity problems.

▶ **Lemma 2.4** (Lemma 2.9 of [48]). *Let $X \times Y$ be a finite joint probability space with measure μ. Let G be the bipartite graph on $X \cup Y$ such that $(x, y) \in X \times Y$ is an edge iff $\Pr[x, y] > 0$ with respect to μ. Assume that G is connected, and let δ be the minimum nonzero probability in the joint distribution. Then, we have that*

$$\rho(X, Y) \leq 1 - \delta^2/2.$$

2.2 Influences

Recall the influence of a function over a probability space.

▶ **Definition 2.5.** Let $X_1, \ldots, X_n$ be finite independent probability spaces, and let $f : X_1 \times \cdots \times X_n \to \mathbb{R}$ be a function. Let $Y_i = X_1 \times \cdots \times X_{i-1} \times X_{i+1} \times \cdots \times X_n$. The influence is

$$\mathrm{Inf}_i(f) = \mathop{\mathbb{E}}_{x \in Y_i}[\mathrm{Var}_{z \in X_i} f(x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_n)].$$

Likewise, we need the notion of low-degree influences. We use the multilinear-polynomial definition used many times previously (e.g., [49, 13, 48]).

▶ **Definition 2.6** (e.g., Definition 3.4, 3.7 of [49]). Let $X_1, \ldots, X_n$ be finite independent probability spaces, and let $f : X_1 \times \cdots \times X_n \to \mathbb{R}$ be a function. For each $i \in [n]$, let q_i be the cardinality of the support of X_i. Let $\alpha_1^{(i)}, \ldots, \alpha_{q_i}^{(i)} : X_i \to \mathbb{R}$ be an orthonormal basis of functions such that $\alpha_1^{(i)} \equiv 1$. Let $\Sigma = [q_1] \times \cdots [q_n]$. Now, f can be uniquely expressed as

$$f = \sum_{\sigma \in \Sigma} c_\sigma \prod_{i=1}^{n} \alpha_{\sigma_i}^{(i)}.$$

for $c_\sigma \in \mathbb{R}$, which we call the Fourier coefficients. For $\sigma \in Q$, let $|\sigma| = |\{i \in [n] \mid \sigma_i \neq 1\}|$. The *low-degree influence* for $d \in [n]$ is

$$\mathrm{Inf}_i^{\leq d} f = \sum_{\sigma \in \Sigma, |\sigma| \leq d, \sigma_i \neq 1} c_\sigma^2.$$

The following is a key elementary fact concerning influences.

▶ **Lemma 2.7** (e.g., Proposition 3.8 [49]). *Consider $f : X_1 \times \cdots \times X_n \to \mathbb{R}$. For all integers $d \geq 1$,*

$$\sum_{i=1}^{n} \operatorname{Inf}_i^{\leq d} f \leq d \operatorname{Var} f.$$

In particular, for all $\tau > 0$, $|\{i \in [n] \mid \operatorname{Inf}_i^{\leq d} f \geq \tau\}| \leq \frac{d \operatorname{Var} f}{\tau}$.

Sometimes, we look at f from the perspective of different marginal distributions. Consider $f : X_1 \times \cdots X_n \to \mathbb{R}$ where the X_i's are independent. Furthermore, assume that each X_i can be written as $X_i = Y_{i,1} \times \cdots Y_{i,\ell_i}$, where these $Y_{i,j}$'s are independent. Then, we let $\operatorname{Inf}_{X_i}^{\leq d} f$ denote the low-degree influence of f in the ith coordinate with respect to the X_i's. Likewise, we let $\operatorname{Inf}_{Y_{i,j}}^{\leq d} f$ be the influence of the (i, j)th coordinate when viewed from the perspective of $f : Y_{1,1} \times \cdots \times Y_{n,\ell_n} \to \mathbb{R}$.

For each (i, j), let $\beta_1^{(i,j)}, \ldots, \beta_{q_{i,j}}^{(i,j)} : Y_{i,j} \to \mathbb{R}$ be an orthonormal basis of functions such that $\beta_1^{(i,j)} \equiv 1$. Note that $q_i = \prod_{j=1}^{\ell_i} q_{i,j}$. Let $\Sigma' = [q_{1,1}] \times \cdots [q_{n,\ell_n}]$. Then, we have that there exist c_σ's such that $f = \sum_{\sigma \in \Sigma'} c'_\sigma \prod_{i=1}^{n} \alpha_{\sigma_i}^{(i)}$. If $\ell_i \leq D$ for all i, then we have the following result

▶ **Lemma 2.8** (e.g., Claim 2.7 [13]). *If $\ell_i \leq D$ for all $i \in [n]$, then we have for all $i, d \in [n]$ that*

$$\operatorname{Inf}_{X_i}^{\leq d} f \leq \sum_{k=1}^{\ell_i} \operatorname{Inf}_{Y_{i,k}}^{\leq Dd} f.$$

Thus, there exists $k \in [\ell_i]$ such that

$$\frac{1}{D} \operatorname{Inf}_{X_i}^{\leq d} f \leq \operatorname{Inf}_{Y_{i,k}}^{\leq Dd} f.$$

Proof. The proof is a straightforward adaptation of the proof of Claim 2.7 in [13]. ◀

For our applications, we only need the case $D = 2$.

2.3 Invariance principles

Like [2], we use the following result on pairwise independent probability spaces.

▶ **Theorem 2.9** (Lemmas 6.6, 6.9 [48]). *Fix $k \geq 3$. For $1 \leq i \leq n$, let $\Omega_i = X_i^{(1)} \times \cdots \times X_i^{(k)}$ be finite pairwise independent probability spaces with probability measure μ_i such that the probability measures corresponding to $\mu_1, \ldots, \mu_n$ are independent. Let δ be the minimum positive probability among all the μ_i. Let*

$$\rho = \max_{1 \leq i \leq n} \rho(X_i^{(1)}, \ldots, X_i^{(k)})$$

and assume that $\rho < 1$. For every $\epsilon > 0$, there exists $\tau(\delta, \epsilon, \rho), d(\delta, \epsilon, \rho) > 0$ such that for any functions $f_1, \ldots, f_k$ where $f_i : X_1^{(i)} \times \cdots \times X_n^{(i)} \to [0, 1]$ if

$$\forall \ell \in [n], |\{i \mid \operatorname{Inf}_{X_\ell^{(i)}}^{\leq d} f_i > \tau\}| \leq 2$$

then

$$\left| \prod_{i=1}^{k} \mathbb{E}[f_i] - \mathbb{E}\left[\prod_{i=1}^{k} f_i\right] \right| \leq \epsilon.$$

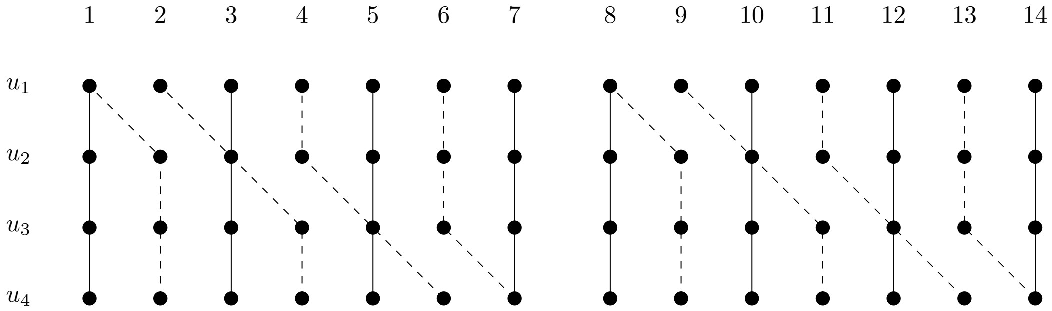


Figure 1 A schematic diagram of the branches for an edge $e = (u_1, u_2, u_3, u_4)$ of V label cover instance Ψ with parameters $k = 4$ and $L = 2$. The ith row represents $\pi_i^{(e)}$ and the jth column represents the input j. The dashed and dotted lines are to indicated the two different branches with the same values with respect to $\pi^{(e)}$. For example, we may deduce from this diagram that $(10, 10, 10, 10)$ and $(9, 10, 11, 11)$ are two branches of e. In particular, we have that $\pi_1^{(e)}(9) = \pi_2^{(e)}(10) = \pi_3^{(e)}(11) = \pi_4^{(e)}(11)$. Note that $\psi_i^{(e)}(j) =\perp$ exactly when the node of the ith row and jth column is at the intersection of two branches. Compare with Figure 1 of [13].

In other words, if the product of the expected values and the expected value of the product significantly differ, then there must exist three functions with a common high low-degree influence coordinate. Note that the number "three" is crucially used in our reduction in Section 4.

3 V label cover

In this section, we propose a variant of hypergraph label cover which seems to plausibly have perfect completeness while also allowing for new hardness reductions. It can be thought of as a generalization of 2-to-1 label cover.

3.1 Definition

Let $k \geq 2$ and $L \geq 1$ be positive integers. An instance of k-uniform V-label cover is a k-uniform hypergraph on vertex set U. The constraints are on k-tuples $E \subseteq U^k$. Each edge $e = (u_1, \ldots, u_k)$ also has projection maps $\pi_1^{(e)}, \ldots, \pi_k^{(e)} : [(2k-1)L] \to [kL]$ with the following special property.

- The maps are surjective, in particular for all $i \in [k]$ and $j \in [kL]$,

$$|(\pi_i^{(e)})^{-1}(j)| = \begin{cases} 1 & i \equiv j \mod k \\ 2 & \text{otherwise} \end{cases}$$

In addition we would like to be able to distinguish between the two labels which map to a common value. To do this, we supplement the projection maps with *distinguishing functions* $\psi_1, \ldots, \psi_k : [(2k-1)L] \to \{0, 1, \perp\}$ such that for all $i \in [k]$, the map $x \mapsto (\pi_i^{(e)}(x), \psi_i(x))$ is injective. Furthermore, if $|(\pi_i^{(e)})^{-1}(\pi_i^{(e)}(x))| = 1$, then we define $\psi_i(x) =\perp$, and otherwise $\psi_i(x) \in \{0, 1\}$. We say that $(t_1, \ldots, t_k) \in [(2k-1)L]^k$ is a *branch* of e if there is $\ell \in [kL]$ and $b \in \{0, 1\}$ such that for all i, $(\pi_i^{(e)}(t_i), \psi_i^{(e)}(t_i))$ equals (ℓ, b) or $(\ell, \perp)$. Note that for each branch, there is exactly one $j \in [k]$ such that $\psi_j^{(e)}(t_j) =\perp$. In fact such such an index satisfies $j \equiv \pi_i^{(e)}(t_i) \mod k$ for all j. We say that i is the *junction* of the branch.

To better understand the setup, see Figure 1.

The goal of V-label cover is to produce a labeling of the vertices $\sigma : U \to [(2k-1)L]$. We say that a hyperedge $e = (u_1, \ldots, u_k)$ is *strongly satisfied* if $(\sigma(u_1), \ldots, \sigma(u_k))$ is a branch. In other words, for all $i, j \in [k]$, $\pi_i^{(e)}(\sigma(u_i)) = \pi_j^{(e)}(\sigma(u_j))$ *and* either $\psi_i^{(e)}(\sigma(u_i)) = \psi_j^{(e)}(\sigma(u_j)) \neq \perp$ or exactly one of $\psi_i^{(e)}(\sigma(u_i)), \psi_j^{(e)}(\sigma(u_j))$ is $\perp$. Another way to express this is that $(\pi_i^{(e)}(\sigma(u_i)), \psi_i^{(e)}(\sigma(u_i)))$ is uniform except for one i for which $\psi_i^{(e)}(\sigma(u_i)) = \perp$ (the meeting point in the 'V' of the two branches).

We say the hyperedge is *weakly satisfied* if for some distinct $i, j \in [k]$, $\pi_i^{(e)}(\sigma(u_i)) = \pi_j^{(e)}(\sigma(u_j))$ *and $\sigma(u_i)$ and $\sigma(u_j)$ are in the same branch.*

We now formally state our conjectured intractability of approximating V label cover. Below we state an "induced" version where in the soundness guarantee, for every labeling, most of the hyperedges within any subset of vertices of density ϵ fail to be weakly satisfied. The induced version is needed for our reduction to hypergraph coloring (this is similar to the α conjecture of [13] which was also defined in an induced form). For our Max k-CSP result, it suffices to assume the soundness condition that at most ϵ fraction of edges are weakly satisfiable. For simplicity, we only state the stronger induced version below.

▶ **Conjecture 3.1** (V label cover–conjecture, induced version)**.** *For all $k \geq 2$ and $\epsilon > 0$, there exists an $L \geq 1$ such that for any k-uniform V label cover instance Ψ on label set L and vertex set U and hyperedge set E, it is NP-hard to distinguish between*

- *YES: There exists a labeling for which every hyperedge is strongly satisfied.*
- *NO: For every labeling and every subset $U' \subset U$ with $|U'| \geq |U|\epsilon$, less than ϵ fraction of the edges in $(U')^k \cap E$ are weakly satisfied by the labeling.*

3.2 Compatibility

Consider a domain size $q \geq 2$, an arity $k \geq 2$, and a predicate $P \subseteq [q]^k$. In order to understand the "V label cover–hardness" of this predicate P, for each edge $e = (u_1, \ldots, u_k)$ of our V label cover instance we seek to construct probability distributions on $[q]^{k \times (2k-1)L}$ such that the marginal distribution of each branch of e is supported by P. We define the notion of *V label cover–compatibility* in order to capture exactly what we need.

▶ **Definition 3.2.** For a predicate $P \subseteq [q]^k$, consider $\mu_1, \ldots, \mu_k$ supported on P^2. For $i, j \in [k]$, let $X_{i,j} \sim [q]^2$ be the marginal distribution of μ_i on the jth coordinates. That is, for all $(a, b) \in [q]^2$,

$$\Pr_{(x',y') \sim X_{i,j}}[(x', y') = (a, b)] = \Pr_{(x,y) \sim \mu_i}[(x_j, y_j) = (a, b)].$$

We call the distributions $\mu_1, \ldots, \mu_k$ a *V label cover–compatible family* if they satisfy the following properties.

1. For all $i \in [k]$, $X_{i,i}$ is uniform on $\{(a, a) \mid a \in [q]\}$.
2. For all $i, j \in [k]$ with $i \neq j$ and $X_{i,j}$ is uniform on $[q]^2$.
3. For all $i \in [k]$, $\rho(\mu_i) < 1$, which we define to be

$$\rho(\mu_i) := \rho(X_{i,1}, \ldots, X_{i,k}).$$

We say that P is *V label cover–compatible* if a V label cover–compatible family $\mu_1, \ldots, \mu_k$ exists.

The reason we have k different distributions is because the two connected branches can intersect in k different rows (see Figure 1).

Property (3) of Definition 3.2 precludes any algebraic structure in our predicate that would permit a polynomial-time algorithm. For example, the uniform distribution on the predicate $\{x \in \mathbb{Z}_2^n \mid x_1 + \cdots + x_n = 0\}$ has correlation 1 and allows for Gaussian-elimination to solve exactly.

3.3 Reduction from V label cover to P-CSP

Let $P \subseteq [q]^k$ be a predicate for $q, k \geq 2$ which is V label cover–compatible with distributions $\mu_1, \ldots, \mu_k$. In this section, we show how to reduce an arbitrary instance of V label cover into an instance of P-CSP, the constraint satisfaction problem where all clauses are of the form $(x_1, \ldots, x_n) \in P$. Furthermore, we assign weights to the clauses of this CSP, in which the weights are determined by these distributions μ_i. This reduction is the starting point for showing the conditional NP-hardness results in Section 4.

Let $\Psi = (U, E, L, \{\pi_i^{(e)}\}_{e\in E, i\in[k]}, \{\psi_i^{(e)}\}_{e\in E,i\in[k]})$ be our instance of k-uniform V label cover. For each $u \in U$, we construct $q^{(2k-1)L}$ variables $x_s^{(u)}$, where $s \in [q]^{(2k-1)L}$. Now, for every edge $e = (u_1, \ldots, u_k) \in E$ and every $s^{(1)}, \ldots, s^{(k)} \in [q]^{(2k-1)L}$ with the following property:

- For any $t_1, \ldots, t_k \in [(2k-1)L]$ such that $(t_1, \ldots, t_k)$ is a branch of e, we have $(s^{(1)}_{t_1}, \ldots, s^{(k)}_{t_k}) \in P$,

we add the constraint $(x^{(u_1)}_{s^{(1)}}, \ldots, x^{(u_k)}_{s^{(k)}}) \in P$. Looking back at Figure 1, we have that any assignment of values from $[q]$ to the nodes of the schematic such that each branch is an element of P corresponds to some choice $(s^{(1)}, \ldots, s^{(k)})$.

Let Φ be the resulting instance. Although we have described the clauses, we have not yet determined the relative weights of the clauses.

▶ **Claim 3.3.** *If Ψ has a labeling $\sigma : U \to [(2k-1)L]$ which strongly satisfies every hyperedge, then we have that Φ has a perfect satisfying assignment. In other words, this reduction has perfect completeness.*

Proof. For each $u \in U$, and $s \in [q]^{(2k-1)L}$, we let $x_s^{(u)} = s_{\sigma(u)}$. One can verify this assignment satisfies Φ. ◀

Now, fix $e = (u_1, \ldots, u_k) \in E$. For each $\ell \in [kL]$, let $(a_1, \ldots, a_k), (b_1, \ldots, b_k)$ be the two branches of e such that $\pi_i^{(e)}(a_i) = \pi_i^{(e)}(b_i) = \ell$ for all i. Let $j \in [k]$ be the unique index for which $a_j = b_j$, (i.e., j is the junction). Let I be the index set $I := \{(i, a_i) \mid i \in [k]\} \cup \{(i, b_i) \mid i \in [k]\}$; note that $|I| = 2k - 1$. Let $\Omega_\ell^{(e)} \sim [q]^I$ be the probability distribution isomorphic to μ_j such that the marginals $x_1, \ldots, x_k, y_1, \ldots, y_k$ of μ_j correspond to the marginals indexed by $(1, a_1), \ldots, (k, a_k), (1, b_1), \ldots, (k, b_k)$ of $\Omega_\ell^{(e)}$.

Let

$$\nu^{(e)} := \prod_{\ell\in[kL]} \Omega_\ell^{(e)},$$

where the product is over independent distributions. Note that the support of $\nu^{(e)}$ can be identified with $[q]^{[k]\times[(2k-1)L]}$ since each $(i, a_i) \in [k] \times [(2k-1)L]$ is accounted for in some branch. We let $Y_j^{(e,i)}$ be the marginal distribution of coordinate $(i, j) \in [k] \times [(2k-1)L]$ of $\nu^{(e)}$. For any $i \in [k]$ and $\ell \in [kL]$, we let $X_{i,\ell}^{(e)}$ be the marginal distribution on the indices

$\{(i,t) \mid \pi_i^{(e)}(t) = \ell\}$. In particular, if i is a junction, the meeting point of the branches, then $Y_t^{(e,i)} = X_{i,\ell}^{(e)}$. Otherwise, $X_{i,\ell}^{(e)}$ is the product of two Y's:

$$X_{i,\ell}^{(e)} = \prod_{t \in (\pi_i^{(e)})^{-1}(\ell)} Y_t^{(e,i)}.$$

This distribution $\nu^{(e)}$ specifies the probability distribution of the clauses corresponding to a particular edge of the label cover instance. These probabilities are the relative weights of the clauses in the instance.

4 Perfect-completeness approximation resistance and Max-k-CSP$_q$

A natural question to ask concerning V label cover is if it reduces to natural families of predicates which are hard to approximate, even when guaranteed perfect completeness. In the case of imperfect completeness, Austrin and Mossel [2] showed assuming the Unique Games Conjecture that if a predicate $P \subseteq [q]^k$, for some finite domain size q, supports a balanced pairwise independent distribution, then P is *approximation resistant*. That is, for all $\epsilon > 0$, it is NP-hard to distinguish between $1-\epsilon$-satisfiable and $\frac{|P|}{q^k} + \epsilon$-satisfiable P-CSPs. Only a few years later, in a breakthrough by Chan [9], unconditional approximation resistance was shown for any P which supports a balanced pairwise independent subgroup. We hope that establishing a similar conditional results for perfect completeness will spur unconditional results in this domain.

In order to reduce from V label cover, we need a more stringent criteria than merely supporting a balanced pairwise independent distribution. We call these more structured distributions *pairwise-independent V label cover-compatible.*

▶ **Definition 4.1.** Let $q \geq 2, k \geq 3$ be parameters. Let $P \subseteq [q]^k$ be a predicate. We say that P is *pairwise–independent V label cover–compatible* if there exists a V label cover–compatible family $\mu_1, \ldots, \mu_k$ supported on P^2 (with marginals $X_{i,j}$, $i, j \in [k]$) with the additional property that:

4. For all $i \in [k]$ and $j \neq j' \in [k]$, we have that $X_{i,j}$ and $X_{i,j'}$ are pairwise independent.

To motivate the definition, one way to view property (4), when combined with properties (1) and (2) of Definition 3.2, is that P does not just support a pairwise independent distribution, but that the distribution can preserve pairwise independence even when conditioning on the value of a coordinate.[8] Assuming the V label cover-conjecture, this property suffices to establish perfect-completeness approximation resistance if we allow what are known as *folded* predicates.[9] Assume that $[q]$ has a + operator (e.g., addition modulo q). We specify that we may use folded versions of our predicate P to be the predicates

$$a \in [q]^k,\ P^{(a)} := \{(x_1 + a_1, \ldots, x_k + a_k) \mid (x_1, \ldots, x_k) \in P\}.$$

Each $P^{(a)}$ has the same cardinality, so incorporating these extra predicates can only increase the severity of the hardness of approximation. Thus, more precisely we say that the *family* of predicates $\{P^{(a)} \mid a \in [q]^k\}$ is perfect-completeness approximation resistant. That is, for

[8] The definition permits a slightly broader class of P (i.e., the distribution can change depending on which coordinate is conditioned on), but our applications will construct P of the type specified in the motivation.

[9] This is a standard assumption in the CSP literature, e.g., [2].

every $\epsilon > 0$, it is NP-hard to distinguish whether a CSP with predicates from $\{P^{(a)} \mid a \in [q]^k\}$ is perfectly satisfiable or is $\frac{|P|}{q^k} + \epsilon$ satisfiable.

▶ **Theorem 4.2.** *Let $P \subseteq [q]^k$ be a predicate which supports a pairwise-independent V label cover–compatible distribution. Then, assuming the V label cover–conjecture, we have that the collection of predicates $\{P^{(a)} \mid a \in [q]^k\}$ is perfect-completeness approximation resistant.*

Proof. The high-level structure of our proof is analogous to that of Austrin and Mossel [2]. The proof proceeds in a couple of stages. First, we describe the reduction from a V label cover instance to an instance of P-CSP, and note that such a reduction preserves perfect completeness. Second, we analyze the soundness of our reduction using Theorem 2.9 to show that if our P-CSP can be well-approximated, then our original V label cover instance also admits an approximation.

Reduction. Let $\Psi = (U, E, L, \{\pi_i^{(e)}\}_{e\in E, i\in[k]}, \{\psi_i^{(e)}\}_{e\in E, i\in[k]})$ be our instance of k-uniform V label cover. Let Φ be the instance of P-CSP guaranteed by the construction in Section 3.3. Let $\nu^{(e)} \in [q]^{[k]\times[(2k-1)L]}$ be the weighting distributions on the clauses corresponding to the hyperedges. Let $\Omega_\ell^{(e)}, X_{i,j}^{(e)}, Y_j^{(e,i)}$ be the marginal distributions described in Section 3.3. By Claim 3.3, our reduction has perfect completeness.

We now modify the CSP Φ into a new CSP Φ' which incorporates folding. For each constraint $(x_{s^{(1)}}^{(u_1)}, \ldots, x_{s^{(k)}}^{(u_k)}) \in P$ and for each $i \in [k]$, let $(s^{(i)})' = s^{(i)} - s_1^{(i)}$ (i.e., subtract $s_1^{(i)}$ from every coordinate). Then, we specify that

$$(x_{(s^{(1)})'}^{(u_1)}, \ldots, x_{(s^{(k)})'}^{(u_k)}) \in P^{(s_1^{(1)}, \ldots, s_1^{(k)})}.$$

One may check that this modification preserves perfect completeness.

Soundness. We view an assignment to Φ' as a collection of functions $\mathcal{F} = \{f_u : [q]^{(2k-1)L} \to [q] \mid u \in U\}$, where $f_u(s)$ is the assigned value for x_s^u. Because of our modification to the CSP, we only specify constraints for $f_u(s)$ when $s_1 = q$. Thus, we may assume that each f_u is *folded*. That is, $f_u(s) + a \equiv f_u(s + (a, \ldots, a)) \mod q$ for all $a \in [q]$. One may check that the f_u's satisfy a clause in Φ' if and only if they satisfy the corresponding clause in Φ. Thus, it is equivalent to focus on the f_u's satisfaction of Φ.

For $a \in [q]$, we let

$$f_u^{(a)}(x) = \begin{cases} 1 & f_u(x) = a \\ 0 & \text{otherwise} \end{cases}$$

We define the influences and low-degree influences (Definitions 2.5 and 2.6) of the $f_u^{(a)}$'s to be with respect to the uniform distribution.

Let $\Phi(\mathcal{F})$ be the fraction of constraints of Φ satisfied by $\mathcal{F}$, using the weights specified by the $\nu^{(e)}$ distributions. We seek to show for any $\epsilon > 0$ if there exists a $\mathcal{F}$ such that $\Phi(\mathcal{F}) > \frac{|P|}{q^k} + \epsilon$, then there exists $\delta > 0$ and $\sigma : U \to [(2k-1)L]$ such that σ weakly satisfies δ fraction of the constraints of Ψ.

It is evident from the construction, that a group of constraints are associated with each $e \in E$. Let $e(\mathcal{F})$ be the fraction of constraints corresponding to ϕ satisfied by $\mathcal{F}$ (that is the measure with respect to $\nu^{(e)}$ of the clauses satisfied by $\mathcal{F}$). We have that

$$\Phi(\mathcal{F}) = \frac{1}{|E|} \sum_{e\in E} e(\mathcal{F}).$$

Thus, if $\Phi(\mathcal{F}) > \frac{|P|}{q^k} + \epsilon$, there exists a subset $E' \subseteq E$ such that $|E'| > (\epsilon/2)|E|$ and $e(\mathcal{F}) \geq \frac{|P|}{q^k} + \epsilon/2$ for all $e \in E'$; as otherwise, $\Phi(\mathcal{F}) \leq \epsilon/2 \cdot 1 + (1-\epsilon/2) \cdot \left(\frac{|P|}{q^k} + \epsilon/2\right) < \frac{|P|}{q^k} + \epsilon$.

Fix, $e = (u_1, \ldots, u_k) \in E'$. Note that

$$\begin{aligned} e(\mathcal{F}) &= \mathop{\mathbb{E}}_{(s_1,\ldots,s_k)\sim\nu^{(e)}}[(f_{(u_1)}(s_1), \ldots, f_{(u_k)}(s_k)) \in P] \\ &= \sum_{r \in P} \mathop{\mathbb{E}}_{(s_1,\ldots,s_k)\sim\nu^{(e)}}[f_{u_1}^{(r_1)}(s_1) \cdots f_{u_k}^{r_k}(s_k)]. \end{aligned}$$

Thus, for some $r \in P$, we have that

$$\mathop{\mathbb{E}}_{(s_1,\ldots,s_k)\sim\nu^{(e)}}[f_{u_1}^{(r_1)}(s_1) \cdots f_{u_k}^{r_k}(s_k)] > \frac{1}{q^k} + \frac{\epsilon}{2|P|}.$$

Let $\epsilon' = \epsilon/(2|P|) > 0$. Also, for all $i \in [k]$, let $\Pi_i^{(e)} = \prod_{\ell=1}^{kL} X_{i,\ell}^{(e)}$. Since each $\Pi_i^{(e)}$ is uniform and f_{u_i} is folded, we have that

$$\mathop{\mathbb{E}}_{s_i \sim \Pi_i^{(e)}}[f_{u_i}^{(r_i)}(s_i)] = \frac{1}{q}.$$

In particular, this implies that

$$\left| \prod_{i=1}^{\ell} \mathbb{E}[f_{u_i}^{(r_i)}(s_i)] - \mathbb{E}\left[\prod_{i=1}^{\ell} f_{u_i}^{(r_i)}(s_i)\right] \right| > \epsilon'.$$

Note that $\nu^{(e)} = \Omega_1^{(e)} \times \cdots \times \Omega_{kL}^{(e)}$ meets the requirements of Theorem 2.9. Thus, there exists $\tau, d > 0$, which are functions of only ϵ' and parameters of $|P|$, such that

$$\exists \ell \in [kL], |\{i : \mathrm{Inf}_{X_{i,\ell}^{(e)}}^{\leq d} f_{u_i}^{(r_i)} > \tau\}| \geq 3.$$

Let $i_1, i_2, i_3 \in [k]$ be three of these coordinates and let $\ell \in [kL]$ be the guaranteed value of ℓ. Observe that we can also write $\Pi_{i_a}^{(e)}$ as

$$\Pi_{i_a}^{(e)} = \prod_{t \in [(2k-1)L]} Y_t^{(e,i_a)}.$$

Note that each $X_{i_a,\ell}^{(e)}$ can be written as the product distribution of at most 2 $Y_t^{(e,i_a)}$'s, where $\pi_{i_a}^{(e)}(t) = \ell$. By invoking Lemma 2.8 with $D = 2$, we have that there exists t_1, t_2, t_3 such that $\pi_{i_a}^{(e)}(t_a) = \ell$ for all $a \in \{1, 2, 3\}$ and

$$\mathrm{Inf}_{Y_{t_a}^{(e,i_a)}}^{\leq 2d} f_{u_{i_a}}^{(r_i)} = \mathrm{Inf}_{t_a}^{\leq 2d} > \frac{\tau}{2},$$

where the equality is due to the fact that the $Y_{t_a}^{(e,i_a)}$ distributions are uniform distributions on $[q]$.

Note that since each 'component' of (e) has two branches, by the Pigeonhole principle, some two of $\{t_1, t_2, t_3\}$ are in the same branch. Thus, any assignment σ for which $\sigma(u_{i_a}) = t_a$ for all $a \in \{1, 2, 3\}$ weakly satisfies e.

For each $u \in U$. Let $S_u \subseteq [(2k-1)L]$ be the set of labels j for which $\mathrm{Inf}_j^{\leq 2d} f_u^{(a)} > \tau/2$ for some $a \in [q]$. Since $\mathrm{Var}\, f_u^{(a)} \leq \max(f_u^{(a)})^2 = 1$, we have by Lemma 2.7 that $|S_u| \leq 4dq/\tau$, which is independent of L. Construct a random labeling $\sigma : U \to [(2k-1)L]$ by sampling each $\sigma(u)$ from S_u independently and uniformly at random (if S_u is empty, let $\sigma(u) = 1$). For each $e \in E'$, we established that there exists $i, i' \in [k]$ and $\ell \in S_{u_i}$ and $\ell' \in S_{u_{i'}}$ such that setting $\sigma(u_i) = \ell$ and $\sigma(u_{i'}) = \ell'$ weakly satisfies e. Thus, in expectation at least

$$\frac{|E'|}{|E|} \cdot \frac{1}{(\max |S_u|)^2} = \frac{\tau^2 \epsilon}{16 d^2 q^2} > 0$$

of the edges are weakly satisfied. Note that this expression is independent of L and the size of Ψ, as desired. ◀

We use this theorem to obtain hardness of approximation results for Max-k-CSP$_q$ when $q \geq 2$ is a prime power. We first need the following lemma which follows from standard constructions in BCH codes.

▶ **Lemma 4.3.** *Let $q \geq 2$ be a prime power, and let $\ell \geq 1$ be odd. There exists $S \subset \mathbb{F}_q^\ell$ with $|S| = q^{(\ell-1)/2}$ such that S is 3-wise linearly independent over $\mathbb{F}_q$. That is, each three-element subset of S is linearly independent.*

▶ Remark. Because of the recent breakthrough that subsets of $\mathbb{Z}_q^n$ which do not have an arithmetic progress of length three have size at most q^{cn} for some $c < 1$, [12, 15], it is impossible to improve that factor of $1/2$ in the exponent to 1 when $q \geq 3$. In particular, Lemma 4.4 can at best be improved to $O_q(k^{2+\gamma})$ for some $\gamma > 0$ (where the O_q notation hides the dependence of q).

▶ **Lemma 4.4.** *For all $q \geq 2$ a prime power and $k \geq 2$, there exists $P \subseteq [q]^k$ which is pairwise–independent V label cover-compatible with $|P| = 2k^3 q^3$.*

The proof is given in Appendix A.

Using the same proof techniques, we have the following corollary.

▶ **Corollary 4.5.** *For $q = 2$ and all $k \geq 2$, there exists $P \subseteq [2]^k$ which is pairwise–independent V label cover–compatible and $|P| = O(k^2)$.*

Proof. Repeat the proof of Lemma 4.4, but note that $S = \{x \in \mathbb{F}_2^\ell : \sum_{i=1}^\ell x_i = 1\}$ is a 3-wise-independent subset of size $2^{\ell-1}$. ◀

Now we may obtain Theorem 1.1.

Proof of Theorem 1.1. The case $q = 2$ follows immediately from Corollary 4.5 and Theorem 4.2. Similarly, if $q \geq 3$ is a prime power, then the result follows from Lemma 4.4 and Theorem 4.2. ◀

This is the first conditional NP-hardness reduction which obtains a soundness of $\frac{\text{poly}(q,k)}{q^k}$ for even one fixed q. Previously, a long code test due to Tamaki and Yoshida [59] obtained $\frac{O(k)}{2^k}$ for when $q = 2$. The currently best known unconditional result for Max-k-CSP$_2$ is $\frac{2^{O(k^{1/3})}}{2^k}$ due to Huang [33]. For $q \geq 3$, the best known result is [30] [46].

Acknowledgments. We would like to thank Elchanan Mossel for useful discussion on a generalization of an invariance principle. We would also like to thank anonymous reviewers for helpful comments.

References

1 Venkat Anantharam, Amin Gohari, Sudeep Kamath, and Chandra Nair. On Maximal Correlation, Hypercontractivity, and the Data Processing Inequality studied by Erkip and Cover. *arXiv:1304.6133 [cs, math]*, April 2013. arXiv: 1304.6133. URL: http://arxiv.org/abs/1304.6133.

2 Per Austrin and Elchanan Mossel. Approximation resistant predicates from pairwise independence. *computational complexity*, 18(2):249–271, 2009. doi:10.1007/s00037-009-0272-6.

3 N. Bansal and S. Khot. Optimal Long Code Test with One Free Bit. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 453–462, October 2009. doi:10.1109/FOCS.2009.23.

4 Nikhil Bansal and Subhash Khot. Inapproximability of hypergraph vertex cover and applications to scheduling problems. In *37th International Colloquium on Automata, Languages and Programming*, pages 250–261, 2010. doi:10.1007/978-3-642-14165-2_22.

5 Vijay V. S. P. Bhattiprolu, Venkatesan Guruswami, and Euiwoong Lee. Approximate hypergraph coloring under low-discrepancy and related promises. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 152–174. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.152.

6 Joshua Brakensiek and Venkatesan Guruswami. New Hardness Results for Graph and Hypergraph Colorings. In Ran Raz, editor, *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 14:1–14:27. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CCC.2016.14.

7 Joshua Brakensiek and Venkatesan Guruswami. The quest for strong inapproximability results with perfect completenes. *Electronic Colloquium on Computational Complexity (ECCC)*, 24(80), 2017. URL: https://eccc.weizmann.ac.il/report/2017/080/.

8 Jonah Brown-Cohen and Prasad Raghavendra. Combinatorial optimization algorithms via polymorphisms. *CoRR*, abs/1501.01598, 2015. URL: http://arxiv.org/abs/1501.01598.

9 Siu On Chan. Approximation resistance from pairwise independent subgroups. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC'13, pages 447–456, New York, NY, USA, 2013. ACM. doi:10.1145/2488608.2488665.

10 Siu On Chan. Approximation resistance from pairwise-independent subgroups. *J. ACM*, 63(3):27, 2016. doi:10.1145/2873054.

11 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Near-optimal algorithms for maximum constraint satisfaction problems. *ACM Trans. Algorithms*, 5(3), 2009. doi:10.1145/1541885.1541893.

12 Ernie Croot, Vsevolod Lev, and Peter Pach. Progression-free sets in Z_{4^n} are exponentially small. *arXiv:1605.01506 [math]*, May 2016. arXiv: 1605.01506. URL: http://arxiv.org/abs/1605.01506.

13 Irit Dinur, Elchanan Mossel, and Oded Regev. Conditional hardness for approximate coloring. *SIAM Journal on Computing*, 39(3):843–873, 2009. doi:10.1137/07068062X.

14 Irit Dinur, Oded Regev, and Clifford D. Smyth. The hardness of 3-uniform hypergraph coloring. *Combinatorica*, 25(5):519–535, 2005.

15 Jordan S. Ellenberg and Dion Gijswijt. On large subsets of F_{q^n} with no three-term arithmetic progression. *arXiv:1605.09223 [math]*, May 2016. arXiv: 1605.09223. URL: http://arxiv.org/abs/1605.09223.

16 Lars Engebretsen. The nonapproximability of non-boolean predicates. *SIAM J. Discrete Math.*, 18(1):114–129, 2004. doi:10.1137/S0895480100380458.

17 Lars Engebretsen and Jonas Holmerin. More efficient queries in pcps for NP and improved approximation hardness of maximum CSP. *Random Struct. Algorithms*, 33(4):497–514, 2008. doi:10.1002/rsa.20226.

18 Vitaly Feldman, Venkatesan Guruswami, Prasad Raghavendra, and Yi Wu. Agnostic learning of monomials by halfspaces is hard. *SIAM J. Comput.*, 41(6):1558–1590, 2012. doi:10.1137/120865094.

19 Hans Gebelein. Das statistische Problem der Korrelation als Variations- und Eigenwertproblem und sein Zusammenhang mit der Ausgleichsrechnung. *ZAMM – Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 21(6):364–379, January 1941. doi:10.1002/zamm.19410210604.

20 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.

21 Venkatesan Guruswami, Johan Håstad, and Madhu Sudan. Hardness of approximate hypergraph coloring. *SIAM Journal on Computing*, 31(6):1663–1686, 2002.

22 Venkatesan Guruswami and Sanjeev Khanna. On the hardness of 4-coloring a 3-colorable graph. *SIAM J. Discrete Math.*, 18(1):30–40, 2004.

23 Venkatesan Guruswami and Euiwoong Lee. Strong inapproximability results on balanced rainbow-colorable hypergraphs. *Combinatorica*, 2015. Accepted.

24 Venkatesan Guruswami and Euiwoong Lee. Nearly optimal NP-hardness of unique coverage. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1724–1730, 2016.

25 Venkatesan Guruswami and Prasad Raghavendra. Constraint satisfaction over a non-boolean domain: Approximation algorithms and Unique-Games hardness. In *Proceedings of the 11th International Workshop on Approximation, Randomization and Combinatorial Optimization (APPROX)*, pages 77–90, 2008. doi:10.1007/978-3-540-85363-3_7.

26 Venkatesan Guruswami, Prasad Raghavendra, Rishi Saket, and Yi Wu. Bypassing UGC from some optimal geometric inapproximability results. *ACM Trans. Algorithms*, 12(1):6, 2016.

27 Venkatesan Guruswami and Ali Kemal Sinop. Improved inapproximability results for maximum k-colorable subgraph. *Theory of Computing*, 9:413–435, 2013. doi:10.4086/toc.2013.v009a011.

28 Gustav Hast. Approximating max k-CSP – outperforming a random assignment with almost a linear factor. In *32nd International Colloquium on Automata, Languages and Programming*, pages 956–968, 2005.

29 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

30 Johan Håstad and Subhash Khot. Query Efficient PCPs with Perfect Completeness. *Theory of Computing*, 1:119–148, September 2005. doi:10.4086/toc.2005.v001a007.

31 H. O. Hirschfeld. A Connection between Correlation and Contingency. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(4):520–524, October 1935. doi:10.1017/S0305004100013517.

32 Sangxia Huang. Approximation resistance on satisfiable instances for predicates strictly dominating parity. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:40, 2012. URL: https://eccc.weizmann.ac.il/report/2012/040/.

33 Sangxia Huang. Approximation resistance on satisfiable instances for sparse predicates. *Theory of Computing*, 10:359–388, 2014. doi:10.4086/toc.2014.v010a014.

34 Sangxia Huang. $2^{(\log N)^{1/10-o(1)}}$ hardness for hypergraph coloring. Technical report, Electronic Colloquium on Computational Complexity (ECCC), 2015. URL: https://eccc.weizmann.ac.il/report/2015/062/.

35 Johan Håstad. On the np-hardness of max-not-2. *SIAM Journal on Computing*, 43(1):179–193, 2014. doi:10.1137/120882718.

36 Gil Kalai. A Fourier-theoretic perspective on the Condorcet paradox and arrow's theorem. *Advances in Applied Mathematics*, 29(3):412–426, 2002. doi:10.1016/S0196-8858(02)00023-4.

37 Sanjeev Khanna, Nathan Linial, and Shmuel Safra. On the hardness of approximating the chromatic number. *Combinatorica*, 20(3):393–415, 2000.

38 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, pages 767–775, 2002.

39 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. doi:10.1137/S0097539705447372.

40 Subhash Khot, Dor Minzer, and Muli Safra. On Independent Sets, 2-to-2 Games and Grassmann Graphs. Technical Report 124, Electrontic Colloquium on Computational Complexity (ECCC), August 2016. URL: https://eccc.weizmann.ac.il/report/2016/124/.

41 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. doi:10.1016/j.jcss.2007.06.019.

42 Subhash Khot and Rishi Saket. Hardness of coloring 2-colorable 12-uniform hypergraphs with ith $2^{(\log n)^{\Omega(1)}}$ colors. In *55th IEEE Annual Symposium on Foundations of Computer Science*, pages 206–215, 2014.

43 Subhash Khot and Rishi Saket. Hardness of finding independent sets in 2-colorable and almost 2-colorable hypergraphs. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1607–1625, 2014.

44 Euiwoong Lee. Hardness of graph pricing through generalized max-dicut. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 391–399, 2015. doi:10.1145/2746539.2746549.

45 Euiwoong Lee. Improved hardness for cut, interdiction, and firefighter problems. *CoRR*, abs/1607.05133, 2016. URL: http://arxiv.org/abs/1607.05133.

46 Konstantin Makarychev and Yury Makarychev. Approximation Algorithm for Non-Boolean Max-k-CSP. *Theory of Computing*, 10:341–358, October 2014. doi:10.4086/toc.2014.v010a013.

47 Colin McDiarmid. A random recolouring method for graphs and hypergraphs. *Combinatorics, Probability and Computing*, 2:363–365, 9 1993. doi:10.1017/S0963548300000730.

48 Elchanan Mossel. Gaussian bounds for noise correlation of functions. *Geometric and Functional Analysis*, 19(6):1713–1756, 2010. doi:10.1007/s00039-010-0047-x.

49 Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. *Ann. of Math. (2)*, 171(1):295–341, 2010. doi:10.4007/annals.2010.171.295.

50 Ryan O'Donnell. Some topics in analysis of boolean functions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 569–578, 2008. doi:10.1145/1374376.1374458.

51 Ryan O'Donnell and Yi Wu. Conditional hardness for satisfiable 3-CSPs. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC'09, pages 493–502, New York, NY, USA, 2009. ACM. doi:10.1145/1536414.1536482.

52 Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 245–254, 2008.

53 Prasad Raghavendra. *Approximating NP-hard problems: Efficient algorithms and their limits.* PhD thesis, University of Washington, 2009.

54 A. Rényi. On measures of dependence. *Acta Mathematica Academiae Scientiarum Hungarica*, 10(3-4):441–451, September 1959. doi:10.1007/BF02024507.

55 S. Sachdeva and R. Saket. Optimal Inapproximability for Scheduling Problems via Structural Hardness for Hypergraph Vertex Cover. In *2013 IEEE Conference on Computational Complexity*, pages 219–229, June 2013. doi:10.1109/CCC.2013.30.

56 A. Samorodnitsky and L. Trevisan. Gowers Uniformity, Influence of Variables, and PCPs. *SIAM Journal on Computing*, 39(1):323–360, January 2009. doi:10.1137/070681612.

57 Alex Samorodnitsky and Luca Trevisan. A PCP characterization of NP with optimal amortized query complexity. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, pages 191–199, 2000. doi:10.1145/335305.335329.

58 Madhu Sudan and Luca Trevisan. Probabilistically checkable proofs with low amortized query complexity. In *39th Annual Symposium on Foundations of Computer Science*, pages 18–27, 1998. doi:10.1109/SFCS.1998.743425.

59 Suguru Tamaki and Yuichi Yoshida. A query efficient non-adaptive long code test with perfect completeness. In Maria Serna, Ronen Shaltiel, Klaus Jansen, and José Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques: 13th International Workshop, APPROX 2010, and 14th International Workshop, RANDOM 2010, Barcelona, Spain, September 1-3, 2010. Proceedings*, pages 738–751, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. doi:10.1007/978-3-642-15369-3_55.

60 Luca Trevisan. Parallel approximation algorithms by positive linear programming. *Algorithmica*, 21(1):72–88, 1998. doi:10.1007/PL00009209.

61 Luca Trevisan. Approximating satisfiable satisfiability problems. *Algorithmica*, 28(1):145–172, 2000. doi:10.1007/s004530010035.

62 Girish Varma. Reducing uniformity in Khot-Saket hypergraph coloring hardness reductions. *arXiv:1408.0262 [cs]*, August 2014. arXiv: 1408.0262. URL: http://arxiv.org/abs/1408.0262.

63 Cenny Wenner. Circumventing d-to-1 for approximation resistance of satisfiable predicates strictly containing parity of width at least four. *Theory of Computing*, 9(23):703–757, 2013. doi:10.4086/toc.2013.v009a023.

64 H. Witsenhausen. On Sequences of Pairs of Dependent Random Variables. *SIAM Journal on Applied Mathematics*, 28(1):100–113, January 1975. doi:10.1137/0128010.

A Proof of Lemma 4.4

Proof. We use a modification of the constructions of [2] and [59]. Let $\ell \geq 3$ be the least odd integer such that $q^{(\ell-1)/2} \geq k$. Thus, $q^\ell \leq k^2 q^3$. View $\mathbb{F}_q^\ell$ as a vector space over $\mathbb{F}_q$. By Lemma 4.3 there exists $S \subset \mathbb{F}_q^\ell$ with $|S| \geq q^{(\ell-1)/2} \geq k$ such that S is 3-wise linearly independent (i.e., every 3-element subset is linearly independent). Let $v^{(1)}, \ldots, v^{(k)} \in S$ be k distinct elements from this set. Define $\langle \cdot, \cdot \rangle$ to be the canonical bilinear form on $\mathbb{F}_q^\ell$. That is, $\langle x, y \rangle = \sum_{i=1}^{\ell} x_i y_i$.

We give an initial attempt to construct our predicate. Let[10]

$$P_0 = \{(\langle v^{(1)}, X \rangle, \ldots, \langle v^{(k)}, X \rangle) : X \in \mathbb{F}_q^\ell\}.$$

We have that $|P_0| \leq q^\ell \leq k^2 q^3$. We show that P_0 satisfies properties (1), (2), and (4). Note that the definition of P_0 defined a natural probability distribution μ. It is clear that μ has

[10] Note that we identify $[q]$ with $\mathbb{F}_q$ in some canonical way.

uniform marginal distributions (since each $v^{(i)}$ is nonzero and X is uniform). Furthermore, the marginal distributions are 3-wise independent (and thus 3-wise uniform) since the $v^{(i)}$'s are 3-wise linearly independent. (We omit the proof, a similar result for pairwise independence is Lemma 4.2 of [2].)

Now, fix $i \in [k]$, define μ_i to be

$$\mu_i := \{x, y \sim \mu \text{ independent} : x_i = y_i\}.$$

Let $X_{i,j}$ with $j \in [k]$ be the marginals of μ_i. We seek to show μ_i satisfies properties (1), (2), and (4). Property (1) follows immediately from the uniform marginals of μ. Now, fix $j \neq i$, since (x_i, x_j) and $(x_i = y_i, y_j)$ are uniform distributions and x_j and y_j are conditionally independent given x_i, we have that

$$\Pr[x_i \wedge x_j \wedge y_j] = \Pr[x_j \wedge y_j | x_i] \Pr[x_i] = \Pr[x_j | x_i] \Pr[y_j | x_i] \Pr[x_i] = \Pr[x_j] \Pr[y_j] \Pr[x_i].$$

Therefore, (x_i, x_j, y_j) is uniform on $\mathbb{F}_q^\ell$. Thus, property (2) and the case $j' = i$ of property (4) follow.

To finish establishing property (4), consider $j \neq j' \in [k] \setminus \{i\}$. We seek to show that $(x_j, x_{j'}, y_j, y_{j'})$ is uniform for which it suffices to show that $(x_i, x_j, x_{j'}, y_j, y_{j'})$ is uniform. Like before,

$$\begin{aligned}&\Pr[x_i \wedge x_j \wedge x_{j'} \wedge y_j \wedge y_{j'}]\\ &\quad = \Pr[x_j \wedge x_{j'} | x_i] \Pr[y_j \wedge y_{j'} | x_i] \Pr[x_i]\\ &\quad = \Pr[x_j] \Pr[x_{j'}] \Pr[y_j] \Pr[y_{j'}] \Pr[x_i] \text{ (3-wise independence of } \mu).\end{aligned}$$

Thus, the μ_i's satisfy properties (1), (2), and (4). Sadly, due to the nice algebraic structure of P_0, we have that $\rho(\mu_i) = 1$ for all i. To rectify this, we create a 'noisy' version of P_0. For $x \in \mathbb{F}_q^k$, let $|x|$ be the number of nonzero coordinates of x. Then, we define P to be

$$P := \{x \in \mathbb{F}_q^k \mid \exists y \in P_0, |x - y| \leq 1\}.$$

Note that $|P| \leq (k+1)|P_0| \leq 2k^3 q^3$. Now, modify the μ_i's to get μ_i''s by the following procedure.

1. Sample $(x, y) \in \mu_i$.
2. Sample $j \in [k]$ and $a, b \in \mathbb{F}_q$ uniformly.
3. If $i = j$, set $x_i = y_j = a$. Otherwise, set $x_i = a$ and $y_j = b$.

Clearly the support of μ_i' is P^2. Also μ_i' preserves properties (1), (2), and (4) of being V label cover-compatible since re-randomizing coordinates can only assist in maintaining pairwise independent distributions.

It remains to show that μ_i' satisfies property (3). The proof of this is similar to that of Lemma 4.6 of [59]. Let

$$Z_{i,j} := \prod_{j=1, j \neq i}^{k} X_{i,j}.$$

It suffices to show that $\rho(X_{i,j}, Z_{i,j}) < 1$. To do that, it suffices to show by Lemma 2.4 that the bipartite graph whose edges are the support of $X_{i,j} \times Z_{i,j}$ is connected. For any $(\alpha, \beta) \in X_{i,j} \times Z_{i,j}$, since with nonzero probability the jth coordinate is rerandomized, we have that $(\alpha', \beta) \in X_{i,j} \times Z_{i,j}$ for all α' in the support of $X_{i,j}$. From this connectivity immediately follows.

Therefore, P has the desired properties. ◀

Scheduling Problems over Network of Machines

Zachary Friggstad*[1], Arnoosh Golestanian[2], Kamyar Khodamoradi[3], Christopher Martin[4], Mirmahdi Rahgoshay[5], Mohsen Rezapour[6], Mohammad R. Salavatipour†[7], and Yifeng Zhang[8]

1 Department of Computing Science, University of Alberta, Edmonton, AB, Canada

2 Department of Computing Science, University of Alberta, Edmonton, AB, Canada

3 Department of Computing Science, University of Alberta, Edmonton, AB, Canada

4 Department of Computing Science, University of Alberta, Edmonton, AB, Canada

5 Department of Computing Science, University of Alberta, Edmonton, AB, Canada

6 Department of Computing Science, University of Alberta, Edmonton, AB, Canada

7 Department of Computing Science, University of Alberta, Edmonton, AB, Canada

8 Department of Computing Science, University of Alberta, Edmonton, AB, Canada

Abstract

We consider scheduling problems in which jobs need to be processed through a (shared) network of machines. The network is given in the form of a graph the edges of which represent the machines. We are also given a set of jobs, each specified by its processing time and a path in the graph. Every job needs to be processed in the order of edges specified by its path. We assume that jobs can wait between machines and preemption is not allowed; that is, once a job is started being processed on a machine, it must be completed without interruption. Every machine can only process one job at a time.

The makespan of a schedule is the earliest time by which all the jobs have finished processing. The flow time (a.k.a. the completion time) of a job in a schedule is the difference in time between when it finishes processing on its last machine and when the it begins processing on its first machine. The total flow time (or the sum of completion times) is the sum of flow times (or completion times) of all jobs. Our focus is on finding schedules with the minimum sum of completion times or minimum makespan.

In this paper, we develop several algorithms (both approximate and exact) for the problem both on general graphs and when the underlying graph of machines is a tree. Even in the very special case when the underlying network is a simple star, the problem is very interesting as it models a biprocessor scheduling with applications to data migration.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases approximation algorithms, job-shop scheduling, min-sum edge coloring, minimum latency

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.5

[*] This research was undertaken, in part, thanks to funding from the Canada Research Chairs program and an NSERC Discovery Grant.

[†] Supported by NSERC.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 5; pp. 5:1–5:18
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Scheduling problems have been studied extensively over the past several decades. In this paper, we consider a class of scheduling problems in which there is an underlying network of machines. Before stating our problem, let us start with the classical job shop scheduling problem. In job shop, we are given a collection J of n jobs and a set M of m machines. Each job j consists of a sequence of μ_j operations $O_{1j}, O_{2j}, \ldots, O_{\mu_j j}$. Operation O_{ij} takes $p_{ij} \in \mathbb{Z}_{\geq 0}$ time units on machine $m_{ij} \in M$. A feasible schedule specifies for each job the times its operations must be performed such that each machine processes at most one operation at any time and for each job, and an operation is performed only if all preceding operations are already performed. We assume all jobs are available at time zero. Let C_j be the completion time of job j in a schedule. Then the makespan of the schedule is $C_{\max} = \max_j C_j$ and the weighted sum of completion time is $\sum_j w_j C_j$ where $w_j \geq 0, j \in J$ are given weights for the jobs. Two common performance measures are to find schedules with minimum makespan or minimum (weighted) sum of completion times. We refer to the latter as min-sum or weighted min-sum objective. When p_{ij}'s are all equal to p_j (i.e. independent of the machine) then we have the *identical machine* setting. Otherwise, we have the *unrelated machine* setting.

There are many special cases of job shop scheduling studied in the literature. One specialization that still generalizes several other problems and has drawn attention more recently is when there is an underlying network of machines. In this setting, we assume we are given a graph $G = (V, E)$ where each edge e corresponds to a machine. Each job $j \in J$ has a specific path Q_j starting at $s_j \in V$ and ending at $t_j \in V$. The path specifies the set of machines the job has to go through in a specific order (i.e. the sequence of its operations). If the graph G is a simple path $P = v_1, v_2, \ldots, v_{m+1}$ (where $v_i v_{i+1}$ corresponds to machine m_i), each $s_j = v_1$ and $t_j = v_{m+1}$ for all jobs $j \in J$ then we get the classical flow shop problem. Another interesting special case is when we have a general graph G, but all p_{ij}'s are 1; this problem becomes the classical packet routing problem in a network (see [14, 15]). There are also works when the underlying graph G is a tree or other special graphs (see [1, 13, 19, 20]).

1.1 Previous work

The amount of previous work on these problems is simply too large to be reviewed comprehensively here. We mention only some of the work and refer the reader to the references in them. Trivial lower bounds used in many of the previous work for makespan are the congestion and dilation lower bounds. If C is the largest congestion of any machine (the maximum over all machines i of the total running time of jobs that have an operation on i) and D is the largest dilation (longest time it would take a job to perform regardless of the presence of other jobs) then $lb = \max\{C, D\}$ is clearly a lower bound on the makespan. For general job shop Shmoys et al. [25] presented an algorithm with performance ratio $O((\log lb)^2 / \log\log lb)$. When jobs can be preempted (i.e. their processing can be paused in the middle of any operations to be resumed later) one can get better results (see [2]).

Acyclic job shop is a special case of job shop where no job has two operations on the same machine. For this setting, Scheideler and Feige [6] present an algorithm to schedule with makespan $O(lb \log lb \log\log lb)$. To complement this, for acyclic job shop with identical machines they provide a family of instances with optimum makespan $\Omega(lb \log lb / \log\log lb)$.

The approximation in [6] is also the best known result for the case of flow shop (which is a special case of acyclic job shop). For the slightly more general setting of flow shop where each job still has to go through the machines in the order they appear but may not need to be run on all of them (i.e. only needs to be run on a subsequence of machines), Mastrolilli and Svensson [18] prove a hardness of approximation of ratio $\Omega(\log^{1-\epsilon} lb)$. For the flow shop problem with identical machines (also referred to as proportionate flow shop), Shakhlevich et al. [23] present a polynomial time algorithm for the weighted min-sum objective.

As mentioned earlier, for the special case of $p_{ij} = 1$ for all i, j, the problem reduces to the packet routing problem, where each job is simply a packet that takes one unit of time to travel each edge (being a machine or a router). For this, the celebrated result of Leighton et al. [14, 15] and subsequent works show that there is a schedule of length $O(lb)$. The most recent result by Harris and Srinivasan [10] show that there exists a schedule of length $7.26 \cdot (C + D)$ (non-constructive) and an algorithm that finds a schedule of length $8.84 \cdot (C + D)$. More recently, Peis et al. [19] have shown that for the case of packet routing on a tree, one can get a schedule of length at most $C + D - 1$; so this implies a simple 2-approximation. For the special case of packet routing when G is simply a path and all packets go from left-to-right, [1, 12] show that the schedule in which at each time step each machine (edge) processes the job that has the shortest distance to go finds the optimum solution for the min-sum objective. Similar algorithms (namely furthest-to-go first) find the optimum solution for makespan objective [12].

For packet routing for in-trees or out-trees (directed trees in which the in-degree of each node is at most one, or out-degree is at most one, respectively) results of [16] show that the furthest-to-go strategy gives optimum solution for makespan. Based on this, [19] observe that it is easy to get a 2-approximation for makespan on undirected trees (by converting the tree into a rooted tree and splitting each schedule into two stages where in the first stage all the packets must first go up and then all the packets must go down to their destination in the 2nd stage). Similar results are claimed by Kowalski et al. [13] for makespan and min-sum objective on trees.[1]

In [17, 22], the authors give a general framework for a broad class of scheduling problems (using LP rounding) that shows that any approximation algorithm with ratio ρ w.r.t. the trivial lower bound lb for makespan can be used to obtain a $2e\rho$ approximation for the min-sum objective. As a special case, this applies to the scheduling problems on networks of identical machines. We will use this result in some of our results. It is worth pointing out that some of the ideas in [17, 22] which are also used in subsequent works have similarities to the ideas of approximation of minimum latency in vehicle routing problems (like the classical minimum latency) which use an approximation for minimum k-stroll or minimum k-spanning tree (k-MST) as a subroutine (see [4] and earlier works).

More recent works have looked at some other variants of scheduling on a network. Im and Moseley [11] look at the online scheduling problem where the network is a tree. In their model, the edges are considered routers and each leaf node corresponds to a machine. Each job must start from the root and then pass through the routers to arrive at a machine to be scheduled on. Each router and machine can process one job at a time. Machines may be unrelated, but routers are identical. They present constant factor competitive approximations using constant speed-up for makespan. Bhattacharya et al. [3] look at coordination mechanism for routing problems on a tree.

1.2 Our results

All of our results are for the identical machines setting (so each job $j \in J$ has a processing time p_j, independent of the machine).

[1] They claim a 3-approximation for makespan, and a 7-approximation for the min-sum objective, but the sketch of the proof they provide for the latter seems incorrect and there is no full proof for it.

Our first result is really just some smaller observations on our part, our more interesting results are mentioned later. However, it points out an improvement for the acyclic job shop problem with identical machines, so we think it bears mentioning.

▶ **Theorem 1.** *For trees, for both makespan and min-sum objective, there are polynomial time $O(\min\{\log n, \log m, \log p_{\max}\})$-approximation algorithms, where $p_{\max}$ is the maximum processing time among all jobs. If all jobs have unit processing time, then there is a polynomial time $4e$-approximation for the min-sum objective.*

For acyclic job shop with identical machines, under both the makespan and the min-sum objective there is an $O(\min\{\log n\ell, \log p_{\max}\})$-approximation where ℓ is the maximum number of machines in a job's sequence.

Note $p_{\max} \leq lb$ so this improves over the approximation for acyclic job shop in [6] by an $O(\log\log lb)$-factor, but only for the identical machines case. Recall that [6] show existence of family of instances of acyclic job shop with identical machines having optimum makespan $\Omega(lb \log lb / \log\log lb)$, so the upper bound is tight within an $O(\log\log lb)$ factor.

We should point out that earlier works [1, 12] imply a 2-approximation for minimizing the makespan for identical jobs on trees. We also consider a special case of trees, called junction-trees: in this setting, the network is a rooted tree T and for each job $j \in J$, the Q_j path for j contains the root. A special junction-tree is when T is simply a star with all the jobs starting and ending at the leaves of T.

▶ **Theorem 2.** *For scheduling on junction-trees, there is a 4-approximation for makespan and a $8e$-approximation for the min-sum objective. Furthermore, if all processing times are 1, there is a different 3-approximation algorithm for the min-sum objective.*

Perhaps the strongest and most technical result of our paper is for the simplest setting of star networks. We prove the following.

▶ **Theorem 3.** *For the min-sum objective on stars where all the jobs start and end on leaves there is a 7.279-approximation algorithm. For the special case of unit processing time, there is a 1.796-approximation algorithm.*

This setting is more interesting than one might initially think; it is closely related to biprocessor scheduling problems studied in, say, [9]. This connection is examined more closely at the start of Section 2.

Another special case of junction trees is when each job starts at the root and may take (any) root-to-leaf node in order to be completed. So there is not a specified path of machines that job j must run on. Instead, we have to decide the path as well as how to schedule the jobs. This is the same setting as in [11] for which the authors present online algorithms. It turns out for this special case computing a schedule with the min-sum objective can, in fact, be solved in polynomial time. We call this problem *rooted-tree routing scheduling.*

▶ **Theorem 4.** *For the rooted-tree routing scheduling, there is a polynomial time algorithm to compute a schedule with the min-sum objective.*

Outline of the paper: We start by studying the simplest setting (star networks) and prove Theorem 3 in Section 2. The approximation algorithms for trees and junction trees as well as the observation for acyclic job shop with identical machines (Theorems 1, 2, and 4) are presented in Section 3.

2 Approximation Algorithms for Stars

In this section, we look at the min-sum objective for scheduling on a star where jobs start/end at leaves. One problem related to the scheduling problem defined on a star network is *biprocessor scheduling* or *data migration* which can be modelled as edge sum-coloring or edge sum multi-coloring [7, 8, 9]. In the data migration problem, one has to move data stored among devices in a network from one configuration to another. The network is modeled as a graph $G = (V, E)$ where each vertex $v \in V$ represents a data storage and an edge $e = v_iv_j$ represents the need to transfer data between v_i and v_j. This transfer may take p_e time units and will keep both v_i and v_j busy for that many steps. A transfer cannot be preemptive (hence, once started must run until completed) and no node v_i can be transferring data to/from more than one other data storage at the same time. So, only data transfer over edges that form a matching can happen concurrently. The goal is to find a schedule for these transfers and minimize the makespan (the time the last transfer completes) or the min-sum objective (the average time the transfers are completed).

This is essentially biprocessor scheduling where the nodes are the processors, the tasks are represented by edges, and each task requires two specific resources (its two end-points) in order to run. When all p_e's are one, minimizing the min-sum objective is equivalent to the min-sum edge coloring of G [9], and it has been studied extensively. In the min-sum edge coloring, one has to find a proper edge coloring $\phi : E \to \mathbb{Z}^+$ that minimizes $\sum_e \phi(e)$. One can think of $\phi(e)$ as the time step in which edge e is scheduled to run on the two processors of its end-points. In the min-sum edge multi-coloring, each edge e has a requirement p_e and one has to assign p_e distinct integers (as colors) to e such that for any two adjacent edges the set of colors assigned to them are disjoint. If one further requires each set of colors to form a consecutive sequence of integers, then those p_e integers can be considered to be the time steps in which task $e = v_iv_j$ is supposed to run on the two processors v_i, v_j. The best approximation algorithm for the min-sum edge coloring is due to Halldorsson et al. [9] who present a configuration LP rounding with ratio 1.8298 and a combinatorial 1.8886-approximation. For biprocessor scheduling with arbitrary processing times p_e, Gandhi et al. [7] give a 7.682-approximation.

The problem we are considering, when restricted to networks of stars is another form of biprocessor scheduling in which each task requires being performed on two specific processors and in a specific order. More formally, suppose that the star $T = (V, E)$ with root/center node r is the network and each job $j \in J$ starts and ends at leaf nodes s_j, t_j, respectively. We first create a directed *demand* graph $H = (V_H, E_H)$ whose vertices correspond to machines (i.e. edges of T) and whose arcs correspond to jobs in J, where each arc $(s_j, t_j) \in E_H$ reflects the fact that job j needs to be processed on machines $\{s_j, r\}$ and then on $\{r, t_j\}$. So, $|V_H| = m$ and $|E_H| = n$. We will use $e_j \in E_H$ to refer to a job $j \in J$.

In this Section, we prove Theorem 3. We start first by presenting the algorithm for the general case which achieves an approximation ratio of 7.279. We then present a modified algorithm that has ratio 1.796 for when all p_j's are 1.

2.1 Approximating stars with general processing times

Our algorithm for both the general and unit processing times has the following general framework which is somewhat similar to the general framework of minimizing latency (see [4] and earlier works) to convert a makespan objective to a min-sum objective. Our algorithm works in stages where in each stage we try to find the maximum number of jobs that can be scheduled subject to a makespan bound B, which is increasing geometrically in each iteration.

```
   Data: Auxiliary graph $H$, a constant $c \in \mathbb{R}^{>0}$ to be fixed later
   Result: A scheduling of the jobs
 1 $\alpha \sim U[0, 1)$
 2 $i \leftarrow 1$
 3 $R_1 \leftarrow E_H$;
 4 while $R_i \neq \emptyset$ do
 5     $t_i \leftarrow c^{i+\alpha}$
 6     Find a $(1.5, t_i)$-proper subset $J_i \subseteq R_i$ (cf. Lemma 6).
 7     Schedule $J_i$ using Proposition 7, starting at the previous iteration's completion
         time.
 8     $R_{i+1} \leftarrow R_i \setminus J_i$
 9     $i \leftarrow i + 1$
10 end
```

Algorithm 1: Approximation for the min-sum scheduling on stars with identical machines.

We show how even a bicriteria approximation for this makespan version of the problem can give a good approximation for the min-sum objective. Most of the work is in finding a good schedule subject to the makespan bound.

Given a schedule, for a subset of jobs $\hat{J} \subseteq J$, we define the *makespan* of $\hat{J}$ as the difference in time between when the last job of $\hat{J}$ finishes processing on its last machine and when the first job of $\hat{J}$ begins processing on its first machine. We also define the *load* of a machine i to be the total processing time of jobs in $\hat{J}$ incident to i in H. Note that the notions of makespan (in our original graph T) and load (in our demand graph H) are closely related. We define (ρ, t)-proper sets of jobs, which will be used in our algorithm.

▶ **Definition 5** ((ρ, t)-proper set)**.** For $\rho \geq 1$ and $t > 0$, we call a subset of jobs $\hat{J} \subseteq J$ a (ρ, t)-proper set if the two following conditions hold:

- $|\hat{J}|$ is at least the size of the maximum subset of J that can be scheduled with a makespan of at most t.
- For each machine i, the total load (congestion) of jobs in $\hat{J}$ that have i as their first machine (called the *in-load* of i) is at most $\rho \cdot t$ and also the load of jobs that have i as their second machine (called the *out-load* of i) is at most $\rho \cdot t$.

We, later on, show how we can build a schedule of jobs in a (ρ, t)-proper subset $|\hat{J}|$ with small makespan *and* small average completion time of those jobs in Proposition 7. Assuming we have an algorithm that can find (ρ, t)-proper sets of jobs for any given t, combined with Proposition 7 we show how we can build an algorithm for the star scheduling problem with the min-sum objective. At each iteration i, we fix a value t_i and do the following: we first find a proper set of remaining jobs with respect to t_i and then, we find a "good" scheduling of these jobs. Algorithm 1 describes the procedure formally. [2]

Before we proceed with the analysis of Algorithm 1, we show how to perform Step 6, i.e. find a proper set of jobs among remaining jobs, and also some details about Step 7.

▶ **Lemma 6.** *There is a polynomial time algorithm that finds a* $(1.5, t)$*-proper set for any* t.

[2] We ideally wish to find the largest set of jobs that can be scheduled at any given time t_i. However, to ensure the tractability of our algorithm, we settle for a proper set as defined instead.

Proof. Let OPT_t be the maximum number of jobs from J that can be scheduled with makespan at most t. First, observe that jobs/edges e in H with $p_e > \frac{t}{2}$ do not appear in any feasible scheduling with a makespan of t as each such job needs to run sequentially on two machines. Remove such jobs from consideration. Let $p_{max} = \max_j p_j$; thus $p_{max} \leq t/2$. We will find a set of jobs $\widehat{J}$ such that the in-load of each machine and the out-load of each machine is at most $t + p_{max} \leq 1.5 \cdot t$ and $|\widehat{J}| \geq OPT_t$.

To find this set, we first consider the problem of picking the maximum number of jobs such that for each machine i the in-load and out-load are at most t. Note the size of this set is at least OPT_t. To find such a set, we round an LP relaxation.

Construct an undirected bipartite graph $\tilde{H} = (\tilde{V_1} \cup \tilde{V_2}, \tilde{E})$ from H: corresponding to every vertex $v \in V_H$ (i.e. for each machine), we create two copies $\tilde{v}_1$ and $\tilde{v}_2$ in $\tilde{V_1}$ and $\tilde{V_2}$, respectively; for every (directed) edge $e = (u, v) \in R_i$ (which corresponds to a job) with $p_e \leq t/2$, we put an undirected edge $\tilde{e} = (\tilde{u}_1, \tilde{v}_2)$ into $\tilde{E}$ and let $p_{\tilde{e}}$ denote the corresponding value p_e. We work with the following LP relaxation:

$$\max \left\{ \sum_{e \in \tilde{E}} x_e : \sum_{e \in \delta_{\tilde{E}}(v)} p_e x_e \leq t \;\forall v \in \tilde{V_1} \cup \tilde{V_2}, \quad x \in [0,1]^{\tilde{E}} \right\}$$

This LP is exactly the LP relaxation for the so-called *demand matching* problem whose study was initiated in [24]. From [24] (which uses an iterated relaxation technique) and the fact that the graph $\tilde{H}$ is bipartite, we can find an integer vector $\overline{x} \in \{0,1\}^{\tilde{E}}$ with $\sum_{e \in \tilde{E}} \overline{x}_e \geq OPT_{LP} \geq OPT_t$ such that $\sum_{e \in \delta_{\tilde{E}}(v)} p_e \cdot x_e \leq t + p_{\max}$. The edges in E corresponding to $e \in \tilde{E}$ with $\overline{x}_e = 1$ forms a $(1.5, t)$-proper set. ◀

We should point out that the $(1.5, t)$-proper set obtained in the proof of Lemma 6 has the property that the in-load and out-load of each node is at most $t + p_{max}$. Now we describe a method that, given such a (ρ, t)-proper set $\hat{J}$ (for any $\rho \geq 1$), returns a schedule of them with a makespan of at most $\rho \cdot t$ and furthermore, the average completion time of each job is small.

▶ **Proposition 7.** *Suppose that $\hat{J}$ is a $(1.5, t)$-proper set as obtained by Lemma 6. There is a scheduling of the jobs in $\hat{J}$ with a makespan of at most $2t + 2p_{max} \leq 3t$. Furthermore, the average completion time of a job in that schedule is at most $\gamma = 2t + p_{max} \leq 2.5t$.*

The algorithm for this proposition is a simple 2-stage one: in the first stage, each machine i processes (in some arbitrary order) those jobs in $\hat{J}$ that have i as their first leg, i.e. are going towards the center of the star where this machine is their first leg. Once all the jobs in $\hat{J}$ have arrived at the center of the star (i.e. have completed their first leg), each machine i starts processing the jobs that have i as their second machine, from smallest to largest processing time. It is straightforward to observe that each stage takes at most $t + p_{max} \leq 1.5t$ units of time to complete; so the total makespan of all jobs is at most $2t + 2p_{max} \leq 3t$.

The proof that the average completion time of each job is at most $2t + p_{max}$ is a bit more involved, and we defer the detailed proof to the full version of the paper. Using this proposition in Step 7, we can turn the $(1.5, t_i)$-proper set found in Step 6 into a schedule for that set with makespan at most $3c^{i+\alpha}$ and average completion time of each job in that set will be $2.5c^{i+\alpha}$.

▶ **Theorem 8.** *Algorithm 1 is a 7.279-approximation algorithm for the min-sum objective on stars when jobs have general processing times.*

Proof. Following the notation of [4], let u_j be completion time of j'th job in our schedule and let c_j^{opt} be the completion time of j'th job in a schedule with the optimum min-sum objective (note that these jobs might not be the same). We would like to bound u_j w.r.t. c_j^{opt}. Assume that $c_j^{opt} = dc^k$ for some $d < c$ and some $k \geq 1$. Based on the value of d with respect to the random variable α in Algorithm 1, two cases arise: i) $d < c^\alpha$ or, ii) $d \geq c^\alpha$. For the first case, note that since in the optimum there is a schedule of j jobs with makespan at most $c_j^{opt} = dc^k < c^{k+\alpha}$, the iteration in which the j'th job is scheduled in our algorithm is at most k. Also, note that the completion time of any job in each iteration i of the previous $k-1$ iterations is at most $\rho c^{i+\alpha}$ where $\rho = 3$ and the average completion time of each job in iteration k (using Proposition 7) is at most $\gamma c^{k+\alpha}$ where $\gamma = 2.5$. Thus:

$$u_j \leq \rho \sum_{\ell=1}^{k-1} c^{\ell+\alpha} + \gamma c^{k+\alpha} \leq \frac{c^{1+\alpha}}{c-1}(\gamma c^k - \rho + (\rho-\gamma)c^{k-1}).$$

Similarly, for when $d \geq c^\alpha$, $c_j^{opt} = dc^k < c^{k+1+\alpha}$. Thus, the j'th job is scheduled no later than iteration $k+1$. Therefore:

$$u_j \leq \rho \sum_{\ell=1}^{k} c^{\ell+\alpha} + \gamma c^{k+1+\alpha} \leq \frac{c^{1+\alpha}}{c-1}(\gamma c^{k+1} - \rho + (\rho-\gamma)c^{k}).$$

In the first case, $\alpha \in [\log_c d, 1)$ and in the second case, $\alpha \in [0, \log_c d)$. By taking the expectation over α over the two cases, one gets

$$\begin{aligned}
\mathbf{E}[u_j] &\leq \int_{\log_c d}^{1} \frac{c^{1+\alpha}}{c-1}(\gamma c^k - \rho + (\rho-\gamma)c^{k-1})d\alpha + \int_{0}^{\log_c d} \frac{c^{1+\alpha}}{c-1}(\gamma c^{k+1} - \rho + (\rho-\gamma)c^{k})d\alpha \\
&= \frac{c}{c-1}\Bigg((\gamma c^k - \rho + (\rho-\gamma)c^{k-1}) \int_{\log_c d}^{1} c^\alpha d\alpha \qquad (1) \\
&+ (\gamma c^{k+1} - \rho + (\rho-\gamma)c^{k}) \int_{0}^{\log_c d} c^\alpha d\alpha\Bigg) \\
&= \frac{c}{\ln c}\left(\gamma dc^k - \rho + (\rho-\gamma)dc^{k-1}\right) \leq \frac{c}{\ln c}(\gamma + \frac{\rho-\gamma}{c})c_j^{opt}.
\end{aligned}$$

Setting $\rho = 3$ and $\gamma = 2.5$, and $c = 2.912$ leads to the approximation ratio of 7.279. ◀

2.2 Refinements for the case of unit processing times

In this section, we modify our general framework to obtain better approximation factors for the case of unit processing times. The main new ingredient of the proof is to use a different algorithm to find (ρ, t)-proper sets instead of Lemma 6. Recall that our general framework works in two steps: first, partition the jobs into disjoint blocks, and second, schedule each block separately. For unit processing time, we follow the same general framework but we use a standard b-matching algorithm for partitioning, and a more careful scheduling algorithm to deal with the jobs of each block. Algorithm 2 describes each stage more formally.

In our algorithm, the procedure b-Matching(b) finds a maximum size b-matching (a subgraph with maximum degree b) in the undirected subgraph obtained from the set of edges in R_i in polynomial time (e.g. [5]).

▶ **Lemma 9.** *For even $b \geq 0$, any b-matching can be partitioned into $\frac{b}{2}$ 2-matchings.*

```
Data: Auxiliary graph $H$, a constant $c \in \mathbb{R}^{>0}$ to be fixed later
Result: A scheduling of the jobs
$\alpha \sim U[0, 1)$
$i \leftarrow 1$
$R_1 \leftarrow E_H$
while $R_i \neq \emptyset$ do
    $t_i \leftarrow 2 \left\lfloor \frac{c^{i+\alpha}}{2} \right\rfloor$
    $J_i \leftarrow$ b-Matching($t_i$)
    Decompose $J_i$ into $\frac{t_i}{2}$ disjoint 2-matchings $J_i^1, J_i^2, \ldots, J_i^{\frac{t_i}{2}}$ (see Lemma 9)
    Schedule jobs in $J_i$ according to Lemma 10
    $R_{i+1} \leftarrow R_i \setminus J_i$
    $i \leftarrow i + 1$
end
```

Algorithm 2: Approximation for the min-sum objective on stars with identical jobs.

This is known for b-regular graphs [21]. It is straightforward to prove the same for graphs with maximum degree b as well. The details appear in full version.

Next, we schedule the jobs in each block. We note that using Vizing's algorithm for edge coloring, we can schedule the jobs in J_i using $t_i + 1$ new time steps (details omitted here), however, in order to obtain a better approximation ratio we do the following. Let $\mathcal{J} = \{J_1, J_2, \ldots, J_\ell\}$ be the partitioning constructed by the algorithm, where J_i is a maximum t_i-matching. Recall that each J_i is further partitioned into slots $J_i^1, J_i^2, \ldots, J_i^{\frac{t_i}{2}}$. Our goal is to find a scheduling of jobs in J_i (for each $i \geq 1$) with small makespan for them and at the same time small average completion time. We show how to find a schedule with makespan t_i for each J_i, $i \geq 2$ (relative to the end of the last group J_{i-1}), and with makespan $t_1 + 1$ for J_1; furthermore, for each J_i the average completion time of the jobs in J_i will be $\frac{t_i+1}{2}$. In the following lemma, we slightly abuse the definition of the makespan within each slot to refer to the number of new time units (in comparison to the previous slot) that is used to schedule its edges.

▶ **Lemma 10.** *Given the partitioning $\mathcal{J}$, there exists a scheduling in which every slot J_i^t has makespan of 2, except for the very first slot J_1^1 which has a makespan of 3. The makespan of each job in J_k will be at most $1 + \sum_{\ell=1}^{k} t_k$. Furthermore, the average completion time of jobs in J_k will be at most $1 + \sum_{\ell=1}^{k-1} t_\ell + \frac{t_k+1}{2}$.*

We only sketch the proof here and defer the details to a full version of the paper.

Proof Sketch. Given that each slot J_k^t accommodates a 2-matching, we first develop a schedule for the first slot of J_1 with a makespan of 3. In doing so, we observe that any 2-matching accommodated in a slot can be modified to a cycle (path) whose vertices alternate between having an in-degree of 2 and an out-degree of 2. By scheduling the jobs of J_1^1 with a makespan of 3, we create one *slack* time unit since every machine processes at most 2 jobs. We then carry this slack time unit to the subsequent slots and schedule the jobs in each J_k^t (except J_1^1) with a makespan of 2. ◀

The proof of the following theorem is analogous to that of Theorem 8, and we defer it to the appendix.

▶ **Theorem 11.** *Algorithm 2 is a* 1.796*-approximation algorithm for the star scheduling problem when jobs have unit processing times.*

3 Scheduling on Trees and General Networks

In this section, we first focus on situations where the topology of the machines is a tree and then on the general acyclic job shop setting. We prove Theorems 1, 2, and 4.

We first recall a result from [17, 22] that shows how to convert an approximation for the makespan objective that is relative to the lower bound $\max\{C, D\}$ into an approximation for the weighted min-sum objective losing only an additional constant factor. Here, C is the congestion and D is the dilation of the input. The statement below paraphrases their result.

▶ **Theorem 12** ([17, 22])**.** *Consider an instance of job shop scheduling with jobs J having weights $w_j \geq 0, j \in J$. Suppose for any $J' \subseteq J$ we can find a schedule of J' in polynomial time having makespan $\gamma \cdot \max\{C(J'), D(J')\}$ where $C(J')$ is the maximum congestion of an edge under jobs J' and $D(J')$ is the dilation of J'. Then in polynomial time, we can find a schedule for all of J where the weighted completion time is at most $2e\gamma$ times the minimum possible weighted completion time.*

When we invoke this, we will simply have proved that for the given instance we can schedule all jobs with makespan bounded by a factor of $\max\{C, D\}$. But it should be obvious that we would get the analogous bound if we restricted to any subset of jobs because that restricted instance falls in the same family of instances we are considering (e.g. on a tree or acyclic job shop with identical machines).

3.1 Proof of Theorem 1

First, note that if all p_j's are 1, then we simply have the packet routing problem in a tree. Peis et al. [19] presented a simple algorithm in this setting that has makespan at most $C + D - 1$ (where C and D are congestion and dilation). This, together with the result of [17, 22], yields a $4e$-approximation for the min-sum objective in unit processing time.

Now, suppose that we have general processing times. We first present an algorithm with the ratio $O(\min\{\log m, \log n\})$ with respect to the two lower bounds of C, D for the makespan. Combined with Theorem 12, this yields the same approximation ratio for the min-sum objective. Finally, we focus on the acyclic job shop and present an $O(\min\{\log n\ell, \log p_{\max}\})$-approximation. This will also provide the $O(\log p_{\max})$ part of the guarantee stated in Theorem 1 for trees.

So, we now focus on trees. Let T be the underlying network. Our plan is to present an $O(\log m)$-approximation, and also an $O(\log n)$-approximation for makespan. We simply return the better of the two. For each, we decompose the problem into a logarithmic number of independent instances, each of which is the union of vertex-disjoint junction-tree instances.

To do this, pick an arbitrary node $v_1 \in T$ as the root (we specify which vertex to pick below) and then partition the jobs into two groups: G_1: those jobs j for which their path Q_j contains node v; and the rest are placed in $J - G_1$. Note that no job in $J - G_1$ ever needs processing on any edge incident with v_1, therefore, each such job is over a subtree of $T - v_1$. We claim that we can always pick v_1 such that the number of jobs in each of the subtrees in $T - v_1$ is at most $n/2$.

▶ **Claim 13.** *Given a tree T with some subpaths $Q_1, \ldots, Q_n$ where each Q_i is a s_i, t_i-path for some $s_i, t_i \in V(T)$ one can always pick a vertex $v \in T$ such that the number of paths that are entirely within any subtree of $T - v$ is at most $n/2$.*

Proof. For every edge $e = uv$, if more than $n/2$ of the paths Q_i are contained entirely in one subtree of $T - e$, direct e toward this subtree. Otherwise, direct e arbitrarily. After directing all edges, there is a node v that has no out-going edge. It should be easy to see v has the required properties. ◀

Trees

Note that we can find a schedule for each of the subtrees of $T - v_1$ independently and run them in parallel. Therefore, we can now solve the problem on each of those subtrees independently. For each such subtree, we pick a node as the root again; all the jobs that contain one of these roots form group G_2 and the rest of jobs belong to $J - G_1 - G_2$, and we do this recursively for each subtree. Since each time, the number of jobs left in a subtree halves, we will have at most $\log n$ iterations and hence we obtain $\sigma \leq \log n$ groups $G_1, G_2, \ldots, G_\sigma$ and each group is the union of independent (i.e. vertex-disjoint) junction-tree instances. Using Theorem 2 we can obtain a 4-approximation for makespan of each group. Running these $\log n$ schedules in any arbitrary order gives an $O(\log n)$-approximation for makespan.

The algorithm for finding an $O(\log m)$-approximation is similar. We only need to pick the root v_1 (and subsequent roots) in such a way that the number of edges (i.e. machines) in each subtree left is at most half the number of edges in the original one. Such a node is commonly called a *centroid* of the tree. Therefore, we obtain $\log m$ groups this way, each of which is a collection of independent junction tree instances. Combining these we get an $O(\min\{\log n, \log m\})$-approximation for the makespan on trees and subsequently the same approximation ratio for min-sum objective function.

Acyclic Job Shop

The approximation we devise for acyclic job shop is really just a sequence of simple observations. Recall we are assuming the processing times are integers, so $p_j \geq 1$ for all jobs j. As in [6], by losing a factor of 2 in $p_{\max}, C$, and D, we assume $p_j = 2^k$ for some $k \in \mathbb{Z}_{\geq 0}$. This is achieved by scaling up all p_j to a power of 2. Observe the optimum solution value at most doubles; we could just double the start times of all operations in an optimum solution. Also, any schedule under these scaled processing times yields a schedule under the original times by using the same start times for each operation.

For each integer $0 \leq k \leq \log_2 p_{\max}$, form the group $B_k = \{j : p_j = 2^k\}$. We can view each group B_k as an instance of acyclic job shop with identical jobs, so by [15] there is a solution with makespan $O(C + D)$. More specifically, we can scale the running times of each job in B_k to be 1, which also scales the congestion and dilation by 2^{-k}. In polynomial time, we can find a schedule for these unit-length jobs with makespan $O(2^{-k} \cdot (C + D))$ [15], so under the original running times 2^k we get a solution with makespan $O(C + D)$.

Finally, we simply concatenate the resulting solutions for these $1 + p_{\max}$ groups to get a solution for all jobs with makespan $O(\log p_{\max} \cdot (C + D))$. As this is an approximation relative to the lower bound $\max\{C, D\}$, we also get an $O(\log p_{\max})$-approximation for the min-sum objective using Theorem 12.

For the $O(\log n\ell)$-approximation, we perform the same bucketing but also form a "small job" group $B_{small} = B_0 \cup B_1 \cup \ldots \cup B_a$ where $a = (\log_2 p_{\max}) - \lceil \log_2 n\ell \rceil$. We round up *all* jobs in B_{small} to have processing time 2^a. We can solve B_{small} trivially by a greedy algorithm that simply ensures no machine is idle if it has an available job to process.

The makespan of this schedule will be at most $2^a \cdot \ell \cdot n$ because there are $\ell \cdot n$ operations in total to be performed between all jobs and at any point of time before all jobs are completed at least one machine will be busy. Note $2^a \cdot \ell \cdot n \leq p_{\max} \leq C + D$. We then solve the remaining $O(\log n\ell)$ buckets $B_{a+1}, \ldots, B_{\log_2 p_{\max}}$ as before and concatenate their schedules for a total makespan of $O(\log n\ell) \cdot (C + D))$. Again, using Theorem 12 this yields an $O(\log n\ell)$-approximation for the min-sum objective.

3.2 Proof of Theorem 2

Recall that in this setting the network of our machines forms a tree T rooted at r and the path Q_j for each job j contains r on its path.

3.2.1 General processing times

In this section, we present a 4-approximation for the makespan on junction trees which is based on the trivial lower bounds of C, D. Again, combined with the result of [17, 22], this implies an $8e$-approximation for the min-sum objective function.

Let L be the value of makespan in an optimum solution. Our algorithm for makespan has two stages: in the first stage each job j moves from s_j to r; in the second stage each job j moves from r to t_j. Clearly, each stage can be completed with makespan at most L. We show how each step can be completed with makespan at most $2L$, and this yields a solution with makespan at most $4L$.

It is easier to describe the algorithm for the 2nd stage first: in this setting, all the jobs are already at the root, and the goal is to send them to their destinations (t_j's). If $u_1, \ldots, u_\sigma$ are children of r, it is enough to focus on the jobs that travel down one arbitrary edge ru_i and describe the algorithm for the subtree rooted at u_i. Suppose we sort the jobs based on their processing times from smallest to largest and start sending them (from the smallest) as soon as ru_i is free. Since each job j starts on its first edge ru_i after jobs that have smaller processing time than j, job j does not encounter delay/waiting other than at the root. Let $p_1 \leq p_2 \leq \ldots \leq p_n$ be the jobs going down ru_i. Then the maximum delay any job encounters (which happens for the last job) is $\sum_{i=1}^{n-1} p_i$ which is at most congestion C. Also, note that once j starts on the first edge, the total time it takes to complete j is exactly $|rt_j| \cdot p_j$. Noting that the largest $|rt_j| \cdot p_j$ is dilation D, all jobs are done after at most D steps, once they have started processing. Therefore, the whole makespan is at most $C + D$ which is at most $2L$.

The algorithm for sending the jobs to the root is almost the same. The best way to describe it is to consider running the same algorithm as if the jobs were supposed to start at the root and each job j is to be sent to its start point s_j. Using the same algorithm as above, all jobs can reach their designated vertex s_j in time at most $2L$. Run this schedule backwards to move all jobs j from s_j to r in time at most $2L$.

3.2.2 Special case of unit processing times

Here, we consider the case of junction trees with unit processing time and present a 3-approximation algorithm for the min-sum objective. Since we have jobs of unit processing time, we can think of the schedule in synchronized setting were in each time step each machine starts processing one job that is available for that machine. We assume each $e = uv$ has two buffers (queues) $b_e(u)$ and $b_e(v)$ at the two ends u, v; $b_e(u)$ will buffer the jobs that arrive at u and want to cross e and $b_e(v)$ will buffer the jobs that arrive at v and want to cross u.

Our algorithm, called Algorithm 3, is very simple; it tries to keep the machines busy. More specifically, at each time step, each machine $e = uv$ (where v is parent of u) performs

```
while there is a job unfinished do
    foreach machine $e = uv$ (with $v$ being parent of $u$) do
        if $b_e(u) \neq \emptyset$ then
            process the first job in $b_e(u)$ and pass it to the next buffer;
        else if $b_e(v) \neq \emptyset$ then
            process the first job in $b_e(v)$ and pass it to the next buffer;
    end
end
```

Algorithm 3: Approximation for the min-sum objective on junction trees with unit processing times.

the following: if there is any job in $b_e(u)$ process the next job from $b_e(u)$ and send it along its path, else if there is any job in $b_e(v)$ then process the next job from $b_e(v)$ and send it along its path, else do nothing. Whenever a job arrives at a machine $e = uv$ from whichever end-point, it enters the corresponding buffer. Essentially, the algorithm keeps the machines busy by processing the jobs that have arrived at them (from either end-point), giving priority to the jobs that are moving towards the root (so they are still in their first leg of their path).

We show that this is a 3-approximation for the min-sum objective, which implies the 2nd part of Theorem 2.

▶ **Theorem 14.** *Algorithm 3 is a 3-approximation for min-sum objective.*

We use $\delta(r)$ to denote the set of machines incident to r. For each edge e let $L(e)$ be the set of jobs whose path contains e and $l(e) = |L(e)|$. Recall that for each job j, Q_j is the unique s_j, t_j path and $|Q_j|$ be the number of machines j needs to be processed on. Let OPT denote an optimum schedule and C_{OPT} the total flow time of OPT. We use C to denote the cost of our solution. In the following two lemmas, we get lower bounds for the optimum. The proof of the first lemma is immediate and the proof of the second is deferred to a full version of this paper.

▶ **Lemma 15.** $C_{\text{OPT}} \geq \sum_j |Q_j|$.

▶ **Lemma 16.** $C_{\text{OPT}} \geq \sum_{e \in \delta(r)} \frac{\ell(e)(\ell(e)+1)}{4} + \frac{n}{2}$

Combining the above two, we obtain the following lower bound for optimum.

▶ **Corollary 17.** $C_{\text{OPT}} \geq \frac{1}{3} \left(\sum_{e \in \delta(r)} \frac{\ell(e)(\ell(e)+1)}{2} + n + \sum_j |Q_j| \right)$

This corollary along with the following lemma implies Theorem 14.

▶ **Lemma 18.** $C \leq \sum_{e \in \delta(r)} \frac{\ell(e)(\ell(e)-1)}{2} + \sum_j |Q_j|$.

We defer the details to a full version of the paper and conclude this section by noting that Algorithm 3 is a 2-approximation for the special case when the machines form a star. This is because by $\sum_{e \in \delta(r)} \ell(e) = 2n$ and $|Q_j| = 2$ the bounds proved in Lemmas 16 and 18 simplify to:

$$C_{\text{OPT}} \geq \sum_{e} \frac{\ell(e)^2}{4} + n \qquad \text{and} \qquad \mathrm{C} \leq \sum_{\mathrm{e}} \frac{\ell(\mathrm{e})^2}{2}. \tag{2}$$

Recall that for this setting our (more complicated) algorithm of Theorem 3 yields a 1.796-approximation.

3.3 Proof of Theorem 4

In this setting, each job j starts at the root and, unlike the previous settings in which a job must be processed on all machines along a given (s_j, t_j) path, it can take any path to reach any leaf node of the tree, while it has a processing time of p_j on every machine. For this case, we show that a simple greedy algorithm finds a schedule with the min-sum objective in polynomial time, hence proving Theorem 4.

Suppose $c_1, \ldots, c_d$ are the children of r. Consider an optimum solution OPT and let J_k be the set of jobs that go down a path starting at edge (machine) rc_k. The following observation is immediate:

▶ **Observation 19.** *In any optimum solution, the following two hold:*

1. *The optimum solution processes the jobs in J_k in the order of their processing time from small to large.*
2. *All the jobs in J_k follow the shortest root-to-leaf path.*

Processing jobs from the smallest to the largest is known as SPT (Shortest Processing Time) rule, and it is known that on a single machine, SPT minimizes total flow time (which means it minimizes the total delay/waiting on one machine). Since using SPT there is no delay on subsequent machines for any job, it immediately implies that the optimum sends jobs down each path using SPT rule.

Let $n_k = |J_k|$ and m_k be the length of the path (number of machines from root-to-leaf) jobs in J_k travel. Suppose that the jobs in J_k from small to large are: $j_k^1, j_k^2, \ldots, j_k^{n_k}$. Since each job $j_k^a \in J_k$ will incur a delay only at the root and the delay is $p_{j_k^1} + p_{j_k^2} + \ldots + p_{j_k^{a-1}}$, and has a path of length m_k of machines to go through, the total flow time of j_k^a is $m_k p_{j_k^a} + \sum_{1 \le i \le a-1} p_{j_k^i}$. Thus, the total flow time of all the jobs in J_k is: $\sum_{1 \le i \le n_k} (m_k + n_k - i) p_{j_k^i}$, and the total flow time of all the jobs in OPT is $\sum_{1 \le k \le d} \sum_{1 \le i \le n_k} (m_k + n_k - i) p_{j_k^i}$. We use $h_k = m_k + n_k$ and call it the "load" of the branch rc_k. The following lemma follows easily.

▶ **Lemma 20.** *In any optimum solution, for any two children $c_k, c_{k'}$ of r with $n_k, n'_k > 0$ we must have: $|m_k + n_k - m_{k'} - n_{k'}| \le 1$. In other words, the difference of loads of any two branches is at most 1.*

Proof. By way of contradiction suppose that OPT is an optimum solution and for two children of r we have $n_k, n'_k > 0$ and $h_k \ge h_{k'} + 2$. Suppose that $J_k = j_k^1, j_k^2, \ldots, j_k^{n_k}$ and $J_{k'} = j_{k'}^1, j_{k'}^2, \ldots, j_{k'}^{n_{k'}}$ are the sequences of the jobs scheduled on branches rc_k and $rc_{k'}$, respectively. Suppose we remove job j_k^1 from branch rc_k and add it in front of the queue $J_{k'}$. The total flow time of the jobs on branch rc_k goes down by $h_k p_{j_k^1}$ and the total flow time of the jobs on branch $rc_{k'}$ goes up by $(h_{k'} + 1) p_{j_k^1}$. So the total net change in flow time is $(-h_k + h_{k'} + 1) p_{j_k^1} < 0$, which contradicts optimality of OPT. ◀

We call a schedule in which the load of any two branches differs by at most 1 an almost balanced schedule. So the above lemma shows every optimum solution is almost balanced. We can also assume w.l.o.g. that in any optimum solution for jobs $n, \ldots, 1$, if job 1 (the smallest job) is removed from the schedule, the remaining schedule is still an almost balanced one. In other words, if J_k is the set of jobs including job 1 and are scheduled on branch rc_k then the load h_k is as big as any other branch load. To see this, suppose that job 1 is scheduled on branch rc_k with $h_k < h_{k'}$ for some other branch $rc_{k'}$ with $n_{k'} > 0$. Let i be the smallest job in $J_{k'}$ and swap 1 and i in the schedule. The net change in the total flow time will be $p_i(h_k - h_{k'}) + p_1(h_{k'} - h_k) < 0$ since $p_1 \le p_i$, which is a contradiction.

These properties suggest the following simple greedy algorithm which we show below finds the optimum solution.

```
Sort the jobs in non-increasing order of their processing time, say $p_n, p_{n-1}, \dots, p_1$;
Let $c_1, \dots, c_d$ be the children of $r$; and $J_i \leftarrow \emptyset$ be the queue of jobs going down
  branch $rc_i$;
Let $m_i$ be the length of shortest root to leaf path from $rc_i$ and $n_i \leftarrow |J_i|$;
$j \leftarrow n$;
while $j \geq 1$ do
    $k \leftarrow \text{argmin}_{1 \leq i \leq d}\{m_i + n_i\}$;
    Schedule job $j$ in front of the queue $J_k$;
    $n_k \leftarrow n_k + 1$;
    $j \leftarrow j - 1$;
end
```

Algorithm 4: Solving the rooted-tree problem.

▶ **Theorem 21.** *The greedy algorithm (Algorithm 4) finds an optimum solution.*

Proof. We prove by backward induction on i that the greedy finds the optimum solution for the set of jobs $n, \dots, i$ for all $n \geq i \geq 1$. The case of $i = n$ is trivial. Let $k \leq n$ be an arbitrary integer and suppose that the greedy partial schedule for jobs $n, \dots, k+1$ is optimum for this set of jobs; call this schedule $\mathcal{S}_{k+1}$ and let $\mathcal{S}_k$ be the greedy schedule after adding job k and $\mathcal{O}_k$ be an optimum schedule for jobs $n, \dots, k$. Let $\mathcal{O}'$ be the schedule for $n, \dots, k+1$ obtained from $\mathcal{O}_k$ by removing job k. Since $\mathcal{S}_{k+1}$ is optimum (by hypothesis), $cost(\mathcal{S}_{k+1}) \leq cost(\mathcal{O}')$. Also, note that both $\mathcal{S}_{k+1}$ and $\mathcal{O}'$ are almost balance and have the same number of jobs. Therefore, if $h_{min}(\mathcal{O}')$ and $h_{min}(\mathcal{S}_{k+1})$ are the minimum loads in $\mathcal{O}'$ and $\mathcal{S}_{k+1}$, respectively, then $h_{min}(\mathcal{O}') = h_{min}(\mathcal{S}_{k+1})$. This implies

$$cost(\mathcal{S}_k) = cost(\mathcal{S}_{k+1}) + p_k(h_{min}(\mathcal{S}_{k+1}) + 1) \leq cost(\mathcal{O}') + p_k(h_{min}(\mathcal{O}') + 1) = cost(\mathcal{O}_k). ◀$$

4 Conclusion

We have presented a number of approximations for special cases of acyclic job shop with identical machines. There are still many interesting questions one could ask.

For example, we tightened the bound between lb and the minimum makespan for acyclic job shop with identical machines by an $O(\log\log lb)$ factor, and now the gap is off by only an $O(\log\log lb)$ factor. Can this be further tightened? Perhaps more interestingly, is the acyclic job shop problem with identical machines hard to approximate within any constant? It may be hard to approximate within $\Omega(\log^{1-\epsilon} lb)$, just like flow shop with unrelated machines [18].

Are we resigned to losing logarithmic factors in trees or can we do better? Note that getting an $O(1)$-approximation for instances of acyclic flow shop with identical machines where the underlying network is a path and each job must follow a subpath is still open.

Finally, the fact that the makespan objective for acyclic job shop is super-constant hard does not necessarily mean its min-sum counterpart is also hard. By way of analogy, min-sum set cover admits a constant-factor approximation while its classic variant minimum set cover (which can be viewed as a makespan version) has a logarithmic hardness of approximation. The problem of getting either further improvements under the min-sum objective or establishing a super-constant hardness are both open.

Acknowledgement. We thank Rohit Sivakumar for preliminary discussions on this topic.

References

1 Antonios Antoniadis, Neal Barcelo, Daniel Cole, Kyle Fox, Benjamin Moseley, Michael Nugent, and Kirk Pruhs. Packet forwarding algorithms in a line network. In *LATIN 2014: Theoretical Informatics – 11th Latin American Symposium, Montevideo, Uruguay, March 31 – April 4, 2014. Proceedings*, pages 610–621, 2014. doi:10.1007/978-3-642-54423-1_53.

2 Nikhil Bansal, Tracy Kimbrel, and Maxim Sviridenko. Job shop scheduling with unit processing times. *Math. Oper. Res.*, 31(2):381–389, 2006. doi:10.1287/moor.1060.0189.

3 Sayan Bhattacharya, Janardhan Kulkarni, and Vahab S. Mirrokni. Coordination mechanisms for selfish routing over time on a tree. In *Automata, Languages, and Programming – 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 186–197, 2014. doi:10.1007/978-3-662-43948-7_16.

4 Kamalika Chaudhuri, Brighten Godfrey, Satish Rao, and Kunal Talwar. Paths, trees, and minimum latency tours. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 36–45, 2003. doi:10.1109/SFCS.2003.1238179.

5 William J. Cook, William H. Cunningham, William R. Pulleyblank, and Alexander Schrijver. *Combinatorial Optimization.* John Wiley & Sons, Inc., New York, NY, USA, 1998.

6 Uriel Feige and Christian Scheideler. Improved bounds for acyclic job shop scheduling. *Combinatorica*, 22(3):361–399, 2002. doi:10.1007/s004930200018.

7 Rajiv Gandhi, Magnús M. Halldórsson, Guy Kortsarz, and Hadas Shachnai. Improved bounds for scheduling conflicting jobs with minsum criteria. *ACM Trans. Algorithms*, 4(1):11:1–11:20, 2008. doi:10.1145/1328911.1328922.

8 Rajiv Gandhi and Julián Mestre. Combinatorial algorithms for data migration to minimize average completion time. *Algorithmica*, 54(1):54–71, 2009. doi:10.1007/s00453-007-9118-2.

9 Magnús M. Halldórsson, Guy Kortsarz, and Maxim Sviridenko. Sum edge coloring of multigraphs via configuration LP. *ACM Trans. Algorithms*, 7(2):22:1–22:21, 2011. doi:10.1145/1921659.1921668.

10 David G. Harris and Aravind Srinivasan. Constraint satisfaction, packet routing, and the lovasz local lemma. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 685–694, 2013. doi:10.1145/2488608.2488696.

11 Sungjin Im and Benjamin Moseley. Scheduling in bandwidth constrained tree networks. In *Proceedings of the 27th ACM on Symposium on Parallelism in Algorithms and Architectures, SPAA 2015, Portland, OR, USA, June 13-15, 2015*, pages 171–180, 2015. doi:10.1145/2755573.2755576.

12 Dariusz R. Kowalski, Eyal Nussbaum, Michael Segal, and Vitaly Milyeykovski. Scheduling problems in transportation networks of line topology. *Optimization Letters*, 8(2):777–799, 2014. doi:10.1007/s11590-013-0613-x.

13 Dariusz R. Kowalski, Zeev Nutov, and Michael Segal. Scheduling of vehicles in transportation networks. In *Communication Technologies for Vehicles – 4th International Workshop, Nets4Cars/Nets4Trains 2012, Vilnius, Lithuania, April 25-27, 2012. Proceedings*, pages 124–136, 2012. doi:10.1007/978-3-642-29667-3_11.

14 Frank Thomson Leighton, Bruce M. Maggs, and Satish Rao. Packet routing and job-shop scheduling in O(congestion + dilation) steps. *Combinatorica*, 14(2):167–186, 1994. doi:10.1007/BF01215349.

15 Frank Thomson Leighton, Bruce M. Maggs, and Andréa W. Richa. Fast Algorithms for Finding O(Congestion + Dilation) Packet Routing Schedules. *Combinatorica*, 19(3):375–401, 1999. doi:10.1007/s004930050061.

16 Joseph Y.-T. Leung, Tommy W. Tam, and Gilbert H. Young. On-line routing of real-time messages. *J. Parallel Distrib. Comput.*, 34(2):211–217, 1996. `doi:10.1006/jpdc.1996.0057`.

17 Wenhua Li, Maurice Queyranne, Maxim Sviridenko, and Jinjiang Yuan. Approximation algorithms for shop scheduling problems with minsum objective: A correction. *J. Scheduling*, 9(6):569–570, 2006. `doi:10.1007/s10951-006-8790-4`.

18 Monaldo Mastrolilli and Ola Svensson. Hardness of approximating flow and job shop scheduling problems. *J. ACM*, 58(5):20:1–20:32, 2011. `doi:10.1145/2027216.2027218`.

19 Britta Peis, Martin Skutella, and Andreas Wiese. Packet routing: Complexity and algorithms. In *Approximation and Online Algorithms, 7th International Workshop, WAOA 2009, Copenhagen, Denmark, September 10-11, 2009. Revised Papers*, pages 217–228, 2009. `doi:10.1007/978-3-642-12450-1_20`.

20 Britta Peis, Martin Skutella, and Andreas Wiese. Packet routing on the grid. In *LATIN 2010: Theoretical Informatics, 9th Latin American Symposium, Oaxaca, Mexico, April 19-23, 2010. Proceedings*, pages 120–130, 2010. `doi:10.1007/978-3-642-12200-2_12`.

21 Julius Petersen. Die theorie der regul aren graphs. *Acta Math.*, 15:193–220, 1891. `doi:10.1007/BF02392606`.

22 Maurice Queyranne and Maxim Sviridenko. Approximation algorithms for shop scheduling problems with minsum objective. *Journal of Scheduling*, 5(4):287–305, 2002. `doi:10.1002/jos.96`.

23 Natalia Shakhlevich, Han Hoogeveen, and Michael Pinedo. Minimizing total weighted completion time in a proportionate flow shop. *Journal of Scheduling*, 1(3):157–168, 1998. `doi:10.1002/(SICI)1099-1425(1998100)1:3<157::AID-JOS12>3.0.CO;2-Y`.

24 F. Bruce Shepherd and Adrian Vetta. The demand matching problem. In *Integer Programming and Combinatorial Optimization, 9th International IPCO Conference, Cambridge, MA, USA, May 27-29, 2002, Proceedings*, pages 457–474, 2002. `doi:10.1007/3-540-47867-1_32`.

25 David B. Shmoys, Clifford Stein, and Joel Wein. Improved approximation algorithms for shop scheduling problems. *SIAM J. Comput.*, 23(3):617–632, 1994. `doi:10.1137/S009753979222676X`.

A Proof of Theorem 11

Proof. Similar to our analysis for the case of general processing times, let u_j be completion time of j'th job in our schedule and let c_j^{opt} be the completion time of j'th job in a schedule with the optimum min-sum objective. Assume $c_j^{opt} = dc^k$ for $d < c$. We consider the two cases where $d < c^\alpha$ and $d \geq c^\alpha$. In the first case, u_j is bounded from above by the amortized bound $1 + \sum_{\ell=1}^{k-1} t_\ell + \dfrac{t_k + 1}{2}$, and in the second case, by the amortized bound $1 + \sum_{\ell=1}^{k} t_\ell + \dfrac{t_{k+1} + 1}{2}$, where $t_\ell = 2\left\lfloor \frac{c^{\ell+\alpha}}{2} \right\rfloor$. Note that the first two terms in both of these bounds correspond to the sum of completion times of all the jobs in previous blocks (Δ_k), and the second term corresponds to the amortized completion time of job j in the last block. Simplifying the bound in the first case, we get

$$\begin{aligned} u_j &\leq c^\alpha + c^{1+\alpha} + \sum_{\ell=2}^{k-1} c^{\ell+\alpha} + \frac{c^{k+\alpha}+1}{2} + 1 - c^\alpha - c^{1+\alpha} + 2\left\lfloor \frac{c^{1+\alpha}}{2} \right\rfloor \\ &= \sum_{\ell=0}^{k-1} c^{\ell+\alpha} + \frac{c^{k+\alpha}}{2} + \frac{3}{2} + \beta_j = c^{k+\alpha}\left(\frac{1}{c-1} + \frac{1}{2}\right) - \frac{c^\alpha}{c-1} + \frac{3}{2} + \beta_j, \end{aligned}$$

where $\beta_j = 2\left\lfloor \frac{c^{1+\alpha}}{2} \right\rfloor - c^{\alpha} - c^{1+\alpha}$. For the second case, we obtain the following:

$$u_j \leq c^{k+1+\alpha}\left(\frac{1}{c-1}+\frac{1}{2}\right) - \frac{c^{\alpha}}{c-1} + \frac{3}{2} + \beta_j.$$

Taking the expectation of u_j over α, we get

$$\begin{aligned}
\mathbf{E}\left[u_j\right] &\leq \int_{\log_c d}^{1}\left(c^{k+\alpha}\frac{c+1}{2(c-1)} - \frac{c^{\alpha}}{c-1} + \frac{3}{2} + \beta_j\right) d\alpha + \\
&\quad \int_{0}^{\log_c d}\left(c^{k+1+\alpha}\frac{c+1}{2(c-1)} - \frac{c^{\alpha}}{c-1} + \frac{3}{2} + \beta_j\right) d\alpha & (3)\\
&= \frac{c+1}{2(c-1)}c^k \int_{\log_c d}^{1} c^{\alpha} d\alpha + \frac{c+1}{2(c-1)}c^{k+1}\int_{0}^{\log_c d} c^{\alpha} d\alpha + \\
&\quad \int_0^1 \left(-\frac{c^{\alpha}}{c-1} + \frac{3}{2} + \beta_j\right) d\alpha & (4)\\
&= \frac{c-1}{\ln c}\cdot\frac{c+1}{2(c-1)}dc^k - \frac{1}{\ln c} + \frac{3}{2} + \int_0^1 \beta_j d\alpha. & (5)
\end{aligned}$$

It remains to bound $\int_0^1 \beta_j d\alpha = \int_0^1 \left(2\left\lfloor \frac{c^{1+\alpha}}{2}\right\rfloor - c^{\alpha} - c^{1+\alpha}\right) d\alpha$. Observe that $\lfloor c^{1+\alpha}/2 \rfloor = \kappa$ where $\kappa \in \{1, \ldots, 6\}$ is such that $1+\alpha \in [\log_c 2\kappa, \log_c 2(\kappa+1))$ for $3 \leq c < \sqrt{14}$. The range for parameter c is chosen with some foresight. Therefore,

$$\begin{aligned}
\int_0^1 2\left\lfloor \frac{c^{1+\alpha}}{2}\right\rfloor d\alpha &= 2\left(\int_0^{\log_c 4-1} 1d\alpha + \int_{\log_c 4-1}^{\log_c 6-1} 2d\alpha + \ldots + \int_{\log_c 12-1}^{1} 6d\alpha\right)\\
&= 22 - 2\log_c 23040.
\end{aligned}$$

Finally,

$$\int_0^1 \beta_j d\alpha = \int_0^1 \left(2\left\lfloor \frac{c^{1+\alpha}}{2}\right\rfloor - c^{\alpha} - c^{1+\alpha}\right) d\alpha = 22 - 2\log_c 23040 - \frac{c-1}{\ln c} - \frac{c(c-1)}{\ln c}.$$

Substituting this value in Equation (5) and simplifying, we get

$$u_j \leq \frac{c_j^{opt}(c+1)}{2\ln c} + \frac{47}{2} - 2\log_c 23040 - \frac{c^2}{\ln c} \leq \frac{c_j^{opt}(c+1)}{2\ln c},$$

where the second inequality holds because $\frac{47}{2} - 2\log_c 23040 - \frac{c^2}{\ln c}$ is a negative term for $c > 0$. For $c = 3.59$, we obtain the claimed approximation ratio of 1.796. ◀

Approximating Incremental Combinatorial Optimization Problems

Michel X. Goemans[1] and Francisco Unda[2]

1 MIT, Cambridge, MA, USA
goemans@math.mit.edu

2 MIT, Cambridge, MA, USA
funda@mit.edu

Abstract

We consider incremental combinatorial optimization problems, in which a solution is constructed incrementally over time, and the goal is to optimize not the value of the final solution but the average value over all timesteps. We consider a natural algorithm of moving towards a global optimum solution as quickly as possible. We show that this algorithm provides an approximation guarantee of $(9 + \sqrt{21})/15 > 0.9$ for a large class of incremental combinatorial optimization problems defined axiomatically, which includes (bipartite and non-bipartite) matchings, matroid intersections, and stable sets in claw-free graphs. Furthermore, our analysis is tight.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.1.6 Optimization, G.2. Discrete Mathematics

Keywords and phrases Approximation algorithm, matching, incremental problems, matroid intersection, integral polytopes, stable sets

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.6

1 Introduction

Usually, in the context of combinatorial optimization, a single solution is sought which optimizes a given objective function. This for example could be designing (or upgrading) a network satisfying certain properties. But the solution might be large, and implementing it may mean proceeding in steps. As the adage says “Rome wasn’t built in a day”. Therefore it becomes important to consider not just the value of the (final) solution, but also the values at intermediate steps. Such incremental models have gained popularity in the last years [7, 1, 9], because of their practical applications to network design problems, disaster recovery, and planning.

As a first approximation to this extra level of complexity, we consider the setting in which we want to evaluate our solution at each time step, and would like to maximize the sum of the values of the intermediate solutions. To formalize this, consider a finite ground set E of q elements, together with a valuation $v : 2^E \to \mathbb{Z}_+$. The valuation function measures some quantity of interest over a subset of E, for example, the size of a maximum matching, the maximum value of an independent set in a matroid, or a maximum flow. Our goal is to find a permutation $\sigma : E \to \{1, \ldots, |E|\}$ that maximizes

$$f(\sigma) = \sum_{i=0}^{q} v\left(\{e \in E : \sigma(e) \leq i\}\right). \tag{1}$$

This is a very general class of problems, which also includes for example scheduling problems, production planning problems and routing problems; such problems typically involve finding a permutation of tasks to perform.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 6; pp. 6:1–6:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Even for simple, polynomially computable valuations v, the problem of finding the best σ might be NP-hard. This applies for example to the situation in which E corresponds to some of the edges of a directed or undirected graph $G = (V, E_0 \cup E)$ with capacities on its edges, and $v(F)$ represents the maximum flow value from s to t (where $s, t \in V$) in the graph $(V, E_0 \cup F)$. The NP-hardness of this incremental problem was shown by Nurre and Sharkey [9], see also Kalinowski et al. [7].

On the tractable side, the incremental problem (1) can be solved efficiently if $v(F)$ represents the weight of a maximum-weight independent subset of F in a matroid M with ground set E. Indeed, an optimum permutation can be obtained from a maximum-weight independent set B for the entire ground set E in the following way. First, order B in order of non-increasing weight followed by all elements of $E \setminus B$ in an arbitrary order. In the case of the incremental spanning tree problem, this was also derived in [6].

In this paper, we consider a class of valuations v which arise naturally from unweighted combinatorial optimization problems, and for which we are able to provide a worst-case analysis of a greedy-like algorithm. This class of valuations is defined axiomatically. First, we require that v takes nonnegative integer values,

(A1): $\forall F \subseteq E : v(F) \in \mathbb{N}$

and is monotonically non-decreasing and can only increase by at most 1 when an element is added:

(A2): $\forall A \subset E, \forall e \in E \setminus A : v(A) \leq v(A \cup \{e\}) \leq v(A) + 1$.

Additionally, we assume that for any k with $v(\emptyset) = \min_F v(F) \leq k \leq \max_F v(F) = v(E)$, there exists a set of cardinality at most k achieving the value k:

(A3): For all $k : v(\emptyset) \leq k \leq v(E)$, $\exists A \subseteq E : |A| \leq k$ and $v(A) = k$.

Consider, for example, an independence system $\mathcal{I}$ on $E_0 \cup E$, i.e. $\mathcal{I} \subseteq 2^{E_0 \cup E}$ and $\mathcal{I}$ is closed under taking subsets. Then if we define $v(F)$ for $F \subseteq E$ as the cardinality of the largest independent subset of $E_0 \cup F$, we can easily see that $v(\cdot)$ satisfies $(A1)$, $(A2)$ and $(A3)$. This generalizes the matroid setting mentioned previously.

We further assume one additional key property, that v satisfies the following *discrete convexity* property:

(A4) Discrete Convexity: $\forall A, C \subseteq E$ with $v(C) - v(A) > 1, \exists B : v(B) = v(A) + 1$ and $|B| - |A| \leq \frac{|C| - |A|}{v(C) - v(A)}$. Furthermore if $A \subset C$ then $A \subset B \subset C$.

This discrete convexity is not satisfied by all independence systems. However, we show in Section 2 that if a certain family of polyhedra is integral then the discrete convexity property is satisfied.

▶ **Theorem 1.** *Let $\mathcal{I} \subseteq 2^{E_0 \cup E}$ be any independence system. Let $P(\mathcal{I}) \subseteq \mathbb{R}^{E_0 \cup E}$ be the convex hull of incidence vectors of all independent sets in $\mathcal{I}$. If for every integer k,*

$$P(\mathcal{I}) \cap \left\{ x \mid \sum_{e \in E_0 \cup E} x_e = k \right\}$$

is integral then $(A4)$ holds for $v : 2^E \to \mathbb{Z}_+$ defined as $v(F) = \max\{|I| \mid I \in \mathcal{I}, I \subset E_0 \cup F\}$.

In Section 2, we show that this holds for example when $\mathcal{I}$ corresponds to the matchings in a (not necessarily bipartite) graph, or to the independent sets common to two matroids, or to the (independent or) stable sets in a claw-free graph. This last example, although more esoteric, is interesting as a complete description of $P(\mathcal{I})$ by linear inequalities is unknown but we can nevertheless rely on the above theorem. The incremental valuation problem in the case of *bipartite* matchings was already considered in [7], where the authors propose

Algorithm 1: Quickest-To-Ultimate for Incremental Valuation

```
Input   : A valuation function $v$ as above.
Output  : A permutation $\sigma$ of $E$
Compute $O \subseteq E$ of minimum cardinality such that $v(O) = v(E)$;
Set $F = \emptyset$;
for $i = 1$ to $v(E) - v(\emptyset)$ do
    Compute $S \subset O \setminus F$ such that $v(F \cup S) \geq v(\emptyset) + i$ and $|S|$ is minimum;
    Set $F = F \cup S$;
end
Output $\sigma$ consistent with how elements were added to S;
```

several approximation algorithms, the best achieving an approximation ratio of 3/4. Other problems falling under the framework discussed here were not considered before.

For any valuation satisfying $(A1) - (A4)$, we provide an approximation algorithm for the incremental valuation problem. For an efficient implementation, we assume that we can compute efficiently (or have oracle access to) the valuation $v(\cdot)$ and we can also find efficiently a minimum cardinality set O with $v(O) = v(E)$. Our algorithm first computes a smallest set $O \subseteq E$ achieving $v(O) = v(E)$, and then starting from $S = \emptyset$ with value $v(\emptyset)$, repeatedly and greedily adds a smallest subset of O to increase $v(S)$ by 1 until all elements of O have been added and then finishes the ordering with the elements of $E \setminus O$. This algorithm is formally described in Section 3 and in Algorithm 1. In Section 3, we present a worst-case analysis of this algorithm:

▶ **Theorem 2.** *For any valuation satisfying* $(A1) - (A4)$*, Algorithm 1 (Quickest-to-Ultimate) is a* γ*-approximation for the incremental valuation problem, where*

$$\gamma = \frac{9 + \sqrt{21}}{15} > 0.9055.$$

The proof of this result is given in Section 4. We also show that the bound of γ is tight in the sense that there are instances of the valuation problem in which the algorithm cannot do better.

2 Problems in this Framework

2.1 Maximum matchings

One of the basic problems that falls in this framework is the *Incremental Matching Problem.* Given a graph $G = (V, E_0 \cup E)$, where E_0 denotes the edges already present at the start, we would like to find an ordering of the edges of E so as to maximize the average size of the maximum matching in E_0 union the edges already selected. This corresponds to the valuation with $v(F) = \mu(E_0 \cup F)$ where $\mu(A)$ equals the size of the maximum matching in the graph (V, A). The bipartite version of this problem is considered in [7], and two different greedy approximation algorithms are presented. The first one, Quickest-Increment, is a locally greedy algorithm that seeks to minimize the number of edges needed to increase the size of the matching by one, until we reach a maximum matching of the entire graph. Kalinovski et al. [7] prove an approximation guarantee of $\frac{2}{3}$ for this algorithm. Their second algorithm, Quickest-to-Ultimate, is globally greedy, in the sense that it first computes a maximum matching of the entire G, and then only adds edges from this matching, in a locally greedy fashion. For this algorithm, [7] prove an approximation bound of $\frac{3}{4}$. In this paper, we generalize this algorithm to a larger class of incremental problems and improve the guarantee to $0.9055\cdots$.

This matching problem, even in the non-bipartite case, fits in the framework discussed here. One can show that this valuation $v(F) = \mu(E_0 \cup F)$ satisfies the discrete convexity property $(A4)$, by considering maximum matchings in A and C, and their symmetric difference and carefully arguing about it. Although this is possible, this does not generalize easily to other problems.

Discrete convexity, however, is easier to argue polyhedrally as we show next.

2.2 Polyhedral characterization for discrete convexity

Let $\mathcal{I} \subseteq 2^{E_0 \cup E}$ be any independence system, and let $v(F) = \max\{|I| : I \in \mathcal{I} \text{ and } I \subseteq E_0 \cup F\}$. Let $P = \operatorname{conv}\{\chi(I) : I \in \mathcal{I}\}$ be the convex hull of all independent sets, and as we will see, we do not necessarily need to know a complete description of P in terms of linear inequalities. We will show Theorem 1 that discrete convexity $(A4)$ holds if, for any integer k,

$$P \cap \{x : x(E \cup E_0) = k\}$$

is integral.

Proof of Theorem 1. For A (resp. C), let I_A (resp. I_C) be a maximum independent subset of $E_0 \cup A$ (resp. $E_0 \cup C$). So, $v(A) = |I_A|$ and $v(C) = |I_C|$. Let $\ell = |I_C| - |I_A|$. Now consider $y = \frac{1}{\ell}\chi(I_C) + (1 - \frac{1}{\ell})\chi(I_A)$. By convexity $y \in P$ and by construction, we have $y(E \cup E_0) = |I_A| + 1$. Thus, $y \in P \cap \{x : x(E \cup E_0) = |I_A| + 1\}$, and by integrality of this polytope, we have that there exists $x = \chi(S) \in P \cap \{x : x(E \cup E_0) = |I_A| + 1\}$ with

$$\begin{aligned} |S \cap E| &= \min\{x(E) : x \in P \cap x(E \cup E_0) = |I_A| + 1\} \\ &\leq y(E) = \frac{1}{\ell}|I_C \cap E| + (1 - \frac{1}{\ell})|I_A \cap E| \\ &\leq \frac{1}{\ell}|C| + (1 - \frac{1}{\ell})|A| = |A| + \frac{|C| - |A|}{v(C) - v(A)}. \end{aligned}$$

Thus $B = S \cap E$ satisfies the first part of the claim in Theorem 1.

Now consider the case in which $A \subseteq C$. Proceeding as before, we get

$$y \in P \cap \{x : x(E \cup E_0) = |I_A| + 1\} \cap \{x : x_e = 0 \ \forall e \in E \setminus C\},$$

and this is again an integral polytope since it is the face of an integral polytope. Now minimizing $x(E \setminus A)$ over

$$P \cap \{x : x(E \cup E_0) = |I_A| + 1\} \cap \{x : x_e = 0 \ \forall e \in E \setminus C\},$$

we get $x = \chi(T) \in P \cap \{x : x(E \cup E_0) = |I_A| + 1\}$ with $T \subseteq E_0 \cup C$ and

$$|T \cap (E \setminus A)| \leq y(E \setminus A) = \frac{1}{\ell}|I_C \cap (E \setminus A)| \leq \frac{|C| - |A|}{v(C) - v(A)}.$$

This means that $B = A \cup (T \cap E)$ is such that $A \subseteq B \subseteq C$,

$$|B| \leq |A| + \frac{|C| - |A|}{v(C) - v(A)}$$

and $v(B) \geq v(T) = |I_A| + 1$. Thus either $v(B) = |I_A| + 1$, or we can eliminate one by one elements of $B \setminus A$ as long as $v(\cdot)$ is not equal to $v(A) + 1$. Eventually, we find a set with the right requirements. ◀

2.3 Maximum stable set in claw-free graphs

A graph $G = (V, E)$ is claw-free if it does not include $K_{1,3}$ (the star on 4 vertices) as an induced subgraph. The line graph of any graph is claw-free, but the converse is not true as there exist claw-free graphs which are not line graphs. Minty [8] and Sbihi [10] have shown that the maximum stable (or independent) set in a claw-free graph is polynomially solvable. When the claw-free graph is a line graph, this extends Edmonds' algorithm [4, 3] for maximum matching, as the maximum matching problem in a graph is equivalent to the maximum stable set problem in its line graph.

By taking the line graph, we can extend the incremental matching problem to an incremental stable set problem in a claw-free graph $G = (V, E)$ in which we are given an initial vertex set V_0 and our task is to choose an ordering of the remaining vertices in $V \setminus V_0$ so to maximize the average size of a maximum stable set in the corresponding prefix. Thus, here $v(F)$ denotes the size of the largest stable set in $G[V_0 \cup F]$. As said before, if the claw-free graph is not a line graph, this is a strictly more general problem than the incremental matching problem.

A complete description of the stable set polytope P for claw-free graphs is still unknown (see, eg, section 69.4a in Schrijver [11]), but we can nevertheless use Theorem 1 to show that (A4) holds (the other conditions (A1), (A2) and (A3) obviously hold).

▶ **Theorem 3.** *Let P be the stable set polytope of a claw-free graph $G = (V, E)$. Then for any integer k, we have that*

$$P \cap \left\{ x \in \mathbb{R}^V \mid \sum_{v \in V} x_v = k \right\}$$

is integral.

Proof. We exploit the known adjacency properties of the stable set polytope (of any graph). Chvátal [2] has shown that two stable sets S_1 and S_2 in G are adjacent vertices in the stable set polytope if and only if their symmetric difference $S_1 \triangle S_2$ induces a connected subgraph of G. When the graph is claw-free, this connected subgraph $G[S_1 \triangle S_2]$ must be a path, and therefore this means that $-1 \leq |S_1| - |S_2| \leq 1$.

Consider any vertex x^* of $P \cap \{x \in \mathbb{R}^V \mid \sum_{v \in V} x_v = k\}$. x^* must lie on a face of dimension at most 1 of P, and therefore must be in the line segment between two adjacent vertices of P. But since the sizes of these stable sets can differ by at most 1, we derive that x^* must be a vertex of P, and integrality follows. ◀

Thus our approximation algorithm result applies to the incremental maximum stable set problem in claw-free graphs.

The adjacency argument in the proof of Theorem 3 generalizes in the sense that Theorem 1 is equivalent to imposing that any pair of adjacent vertices of $P(\mathcal{I})$ differ in cardinality by at most one unit.

2.4 Matroid intersection

Another generalization of the incremental version of the bipartite matching problem is to consider the incremental version of matroid intersection. Let M_1 and M_2 be matroids defined on the same ground set, say $E_0 \cup E$, and for $F \subseteq E$, let $v(F)$ be the cardinality of the largest common independent set to the two matroids within $E_0 \cup F$.

For matroid intersection, we can directly use Theorem 1 to show that the discrete convexity holds. The matroid intersection polytope P has been characterized by Edmonds [5], and the integrality of $P \cap \{x | \sum_i x_i = k\}$ follows simply by truncating both matroids to size k. Thus Theorem 1 can be used to prove (A4) and Theorem 2 can be used to derive a better than 0.9-approximation algorithm for the incremental maximum matroid intersection problem.

3 Quickest-To-Ultimate for Incremental Problems

Algorithm Quickest-To-Ultimate (Q2U in short, see Algorithm 1) was introduced by Kalinowski et al. [7] for the problem of incremental flows (defined in the introduction). The general idea behind this algorithm is to reach the maximum valuation possible in the shortest number of steps. In the setting of incremental flows, finding the smallest set O of edges whose addition gives a maximum flow is a hard problem, and they resort to a mixed integer program for finding O. In this direction it is known that the incremental flow problem is NP-hard even if the capacities are restricted to be one or three [9]. In the case of unit capacities, Q2U becomes a polynomial approximation algorithm, and in [7] it is shown that it finds a solution with at least half the value of the optimum for the incremental flow problem with unit capacities, and they also show a matching family of examples in which this approximation ratio is attained as the size of the graph grows. In the case of bipartite matchings, a further restriction of the problem, Q2U is shown to find a solution to the incremental matching problem of value at least 3/4 of the optimum, and they show an instance in which the value obtained by Q2U is $\frac{68}{69}$ of the value attained by the optimal solution. Theorem 2 and the example given in Section 3.1 below close this gap for a more general class of valuations, which includes the incremental matching problem.

We show Theorem 2, that for any valuation satisfying (A1)-(A4) the performance guarantee of the algorithm is at least $\frac{9+\sqrt{21}}{15} > 0.9055$. The proof appears later in this section, and our analysis is tight as we show next.

3.1 Bad instance for Quickest-to-Ultimate

In the special case of matchings, and even bipartite matchings (or any setting which includes bipartite matchings), the analysis is tight. Consider, indeed, a graph formed by a disjoint copies of P_3, a path with 3 edges, and one copy of P_{4b+3}, a path with $4b+3$ edges, see Figure 1. The edges of E_0 correspond to each middle edge in the copies of P_3, and every fourth edge of P_{4b+3}, starting with the second one. The remaining edges are edges of E. The valuation is $v : E \to \mathbb{N}$, where $v(S)$ is the size of a maximum matching using edges from $E_0 \cup S$.

In this graph, we have $q := |E| = 2a + 3b + 2$ edges to be added, the original matching has size $m := a + b + 1$, and it can grow by $r := a + b + 1$. The minimum number of edges we need to add to reach this maximum matching is $c_r := 2a + 2b + 2$. Quickest-to-Ultimate adds these edges in pairs that increase the matching, so it adds two edges for each increment in the maximum matching. The value it attains is then $f_{alg} = (q + 1)(m + r) - \sum_{i=1}^{a+b+1} 2i = 3a^2 + 5b^2 + 8ab + 7a + 9b + 3$. On the other hand, here is a better solution. The solution first adds the b edges of P_{4b+3} that increase the maximum matching from m to $m + b$, then it adds a pairs of edges to increase the matching by a and then it adds $2b + 2$ edges to increase the matching by one. This gives a value f with $f_{opt} \leq f = (q+1)(m+r) - \sum_{i=1}^{b} i - \sum_{i=b+1}^{a+b} (2(i - b) + b) - (2a + b + 2b + 2) = 3a^2 + \frac{11}{2}b^2 + 9ab + 7a + \frac{17}{2}b + 4$. A straightforward optimization over a and b, yields that the minimum value of $\frac{f_{alg}}{f_{opt}}$ is attained when a and b

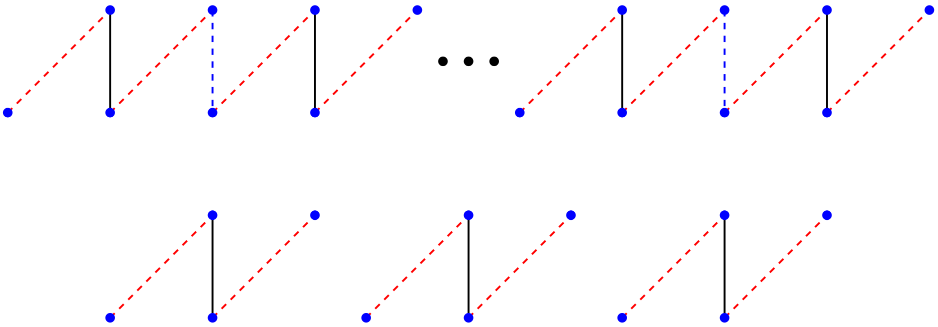


Figure 1 Solid black edges are edges of E_0 and dashed edges are the edges of E.

go to infinity, with $a = \delta b$, with $\delta = \frac{\sqrt{21}}{6} - \frac{1}{2}$, with value of $\frac{9+\sqrt{21}}{15}$. This is the worst case for Q2U and matches the bound we prove in Theorem 2.

4 Analysis

Before diving into the analysis of Q2U, we introduce some notation and exhibit some convexity properties of various sequences associated with these incremental problems.

We denote $v(\emptyset)$ by m, $v(E)$ by $m+r$, and $|E|$ by q. For any permutation σ of E, define

$$d_i(\sigma) := |\{j \in \{0,\ldots,q\} : v(\{e \in E : \sigma(e) \leq j\}) \leq m+i-1\}|,$$

which is the number of elements needed for the solution σ to get to a valuation $m+i$. We will denote by d_i^* the values of $d_i(\sigma^*)$ for an optimal solution σ^* to (1), and by d_i the values of $d_i(\sigma)$ for the permutation σ output by Q2U.

Define for each $i \in \{0,\ldots,r\}$

$$c_i := \min\{|S| : v(S) \geq m+i\}.$$

By definition, we have $c_i \leq d_i$ and similarly $c_i \leq d_i^*$. Also by $(A2)$ we must have $c_i \geq i$, and by $(A3)$,

$$c_i \leq m+i, \tag{2}$$

for $i \in \{0,\ldots,r\}$.

We show that our assumptions imply that both the sequence $\{c_i\}_{i=1}^r$, and the sequence $\{d_i\}_{i=1}^r$ satisfy a convexity property.

▶ **Lemma 4.** *The sequence $\{c_i\}_{i=1}^r$ satisfies*

$$c_{i+1} - c_i \geq c_i - c_{i-1}, \quad 1 \leq i \leq r-1.$$

Proof. To see this, apply $(A4)$ to the respective solutions S_{i-1}, S_i, S_{i+1} where

$$S_j = \arg\min_S \{|S| : v(S) \geq m+j\}.$$

Note first that by $(A2)$ we have that $v(S_j) = m+j$ for $j = i-1, i, i+1$. This implies $v(S_{i+1}) - v(S_{i-1}) = 2 > 1$, and so by $(A4)$, there exists B such that $v(B) = v(S_{i-1})+1 = m+i$ and $2(|S_i| - |S_{i-1}|) \leq 2(|B| - |S_{i-1}|) \leq (|S_{i+1}| - |S_{i-1}|)$. This implies Lemma 4. ◀

The solution given by Q2U also satisfies this same convexity property.

▶ **Lemma 5.** *The sequence* $\{d_i\}_{i=1}^r$ *corresponding to Q2U satisfies*

$$d_{i+1} - d_i \geq d_i - d_{i-1}, \quad 1 \leq i \leq r-1.$$

Proof. To see this, denote by S_i the set computed in the inner loop of the algorithm at step i. That is $S_i \subset F_r \setminus (S_1 \cup \ldots \cup S_{i-1})$ such that $|S_i|$ is minimum and $v(S_1 \cup \ldots \cup S_i) \geq m+i$. Minimality of $|S_i|$, and property $(A2)$ imply that $v(S_1 \cup \ldots \cup S_i) = m+i$, and then $d_i = |S_1 \cup \ldots \cup S_i|$. Now take $i \in \{1, \ldots, r-1\}$. Then $v(S_1 \cup \ldots \cup S_{i+1}) - v(S_1 \cup \ldots \cup S_{i-1}) = 2 > 1$, and then by property $(A4)$ there is a B such that $v(B) = m+i$ and $2(|B| - d_{i-1}) \leq d_{i+1} - d_{i-1}$. Finally by minimality of $|S_i|$ we must have $d_i \leq |B|$, which implies the claim. ◀

We could also show that any optimum ordering σ^* satisfies the same convexity property: $d^*_{i+1} - d^*_i \geq d^*_i - d^*_{i-1}$ for all i, although we will not need this. This requires the second part of (A4) which says that if $A \subseteq C$ then B can be chosen to be sandwiched by A and C.

4.1 Local minima

We also show that the convexity property (A4) implies a relationship between local and global optima, that will be used to derive the optimal upper bound for the Quickest-To-Ultimate Algorithm 1.

▶ **Lemma 6.** *Let S and T be two subsets of E, such that $v(S) = m + |S|$ and $v(T) = m + |T|$, and let S be maximal with this property, that is for any $S' \supset S$ we have $v(S') < m + |S'|$. Then,* $2|S| \geq |T|$.

This is a generalization of the well-known result that any maximal matching is at least half the size of a maximum matching.

Proof. If $|S| \leq |T|$, there is nothing to prove, so we assume that $|T| > |S|$. Now use (A4) with $A = S$ and $C = S \cup T$. Then there is a set B with $S \subset B \subset S \cup T$ such that $v(B) = v(S) + 1$ and

$$|B| - |S| \leq \frac{|S \cup T| - |S|}{v(S \cup T) - v(S)} \leq \frac{|T|}{|T| - |S|}.$$

On the other hand, by the maximality of S, we must have $|B| - |S| \geq 2$. Putting these two together yields

$$2 \leq \frac{|T|}{|T| - |S|},$$

or equivalently

$$2|S| \geq |T|.$$ ◀

4.2 Quickest-To-Ultimate

To analyze Quickest-To-Ultimate, we need to introduce some additional parameters related to the instance being considered. Define

$$p = \max\{|P| : P \subset E, v(P) = m + |P|\},$$

the maximum size of a set that, if added sequentially, increases the valuation at each step. In other words, $p = \max\{i : c_i = i\}$. Clearly $p \leq r$. Also, by maximality of p, we have that

$$\begin{cases} c_i = i & i \leq p \\ c_i \geq p + 2(i-p) & i > p. \end{cases} \tag{3}$$

For Q2U, we are interested in the quantity s, the number of times the set S in the inner loop is a singleton. Note that by Lemma 5, these s iterations occur at the beginning, so an equivalent way to define s is

$$s = \max\{i \in \{1, \ldots, r\} \,:\, d_i = i\}.$$

Our objective is to relate the quantities s and p, which will give us some control over the approximation ratio f_{alg}/f_{opt}. We must have $s \leq p$, since otherwise it would contradict the maximality of p. To get a lower bound on s, define S to be the set of the first s elements added by the algorithm, and $T = P \cap O$, where O is the set of elements chosen by Q2U to first reach $v(E)$ and P is a set of p elements with $v(P) = m + p$. Note that we have $v(S) = m + |S|$ and $v(T) = m + |T|$, and S must be maximal, by definition of s. Then by using Lemma 6, we conclude that $2|S| \geq |T| = |P \cap O|$. This implies that

$$q = |E| \geq |O \cup P| = |O| + |P| - |O \cap P| \geq c_r + p - 2s. \tag{4}$$

And we also know that

$$q \geq c_r. \tag{5}$$

Finally, we need the following inequality. Observe that all the elements that are used by the algorithm come from O, and conversely, all the elements of O must be used by the algorithm to reach valuation $v(E)$, by minimality of c_r. This means that $|O| = c_r = d_r = \sum_{i=1}^{r}(d_i - d_{i-1})$, and using the definition of s, then $c_r = s + \sum_{i=s+1}^{r}(d_i - d_{i-1})$, from which it follows that

$$c_r \geq 2r - s. \tag{6}$$

We are now ready to prove Theorem 2.

Proof of Theorem 2. For any permutation σ, we can rewrite $f(\sigma)$ as

$$f(\sigma) = (q+1)(m+r) - \sum_{i=1}^{r} d_i(\sigma). \tag{7}$$

In particular, for the optimum permutation σ^* and its optimum value $f_{opt} = f(\sigma^*)$, we have:

$$f_{opt} = (q+1)(m+r) - \sum_{i=1}^{r} d_i^* \leq (q+1)(m+r) - \sum_{i=1}^{r} c_i. \tag{8}$$

Using (3) and distinguishing between $i \leq p$, $p < i < r$ and $i = r$, we can write:

$$f_{opt} \leq (q+1)(m+r) - p^2/2 - p/2 + pr - r^2 + r - c_r. \tag{9}$$

Now, denoting the value obtained by Q2U as f_{alg}, and using the definition of s, we have

$$f_{alg} = (q+1)(m+r) - \sum_{i=1}^{r} d_i = (q+1)(m+r) - \sum_{i=1}^{s-1} i - \sum_{i=s}^{r} d_i. \tag{10}$$

To upper bound the last term of (10), we use the following lemma, whose proof is given in the appendix.

▶ **Lemma 7.** *Let $f : \{0, \dots, a\} \to \mathbb{N}$ be a discrete convex function, i.e. $f(i+1) - f(i) \geq f(i) - f(i-1)$, such that $f(0) = 0$ and $f(a) = b$. Furthermore let $b = ka + t$, where $t \in \{0, \dots, a-1\}$. Then,*

$$\sum_{i=0}^{a} f(i) \leq (b + ka^2 + t^2)/2 = \frac{(a+1)b}{2} - \frac{t(a-t)}{2}.$$

Applying this to $f(i) = d_{s+i} - s$, we obtain

$$f_{alg} \geq (q+1)(m+r) - s(s-1)/2 - (r-s+1)s - (r-s+1)(c_r - s)/2 + t(r-s-t)/2,$$

where $t = c_r - s \bmod r - s$. Or after simplification:

$$f_{alg} \geq (q+1)(m+r) - rs/2 - (r-s-1)c_r/2 + t(r-s-t)/2. \tag{11}$$

We need to find the minimum value attainable by $f_{alg}/f_{opt} \leq 1$, which is a lower bound on the approximation ratio. We will show that this lower bound coincides with the upper bound given by the example in Section 3.1. Denote by P_{opt} (resp. P_{alg}) the right-hand-side of inequality (9) (resp. (11)). To find this lower bound, we maximize P_{opt}/P_{alg} over all integral q, m, c_r, r, p, s and t satisfying the conditions:

1. $r \geq p \geq s \geq 0$
2. (5): $q \geq c_r$
3. (4): $q \geq c_r + p - 2s$
4. (6): $c_r \geq 2r - s$
5. $m + r \geq c_r$
6. $t = c_r - s \bmod r - s$.

We first show that the we can ignore all but the quadratic terms in the variables q, m, r, p, s, t. If we double each of q, m, c_r, r, p, s, then t also doubles by **6**, and all inequalities **1**-**5** are still satisfied. Furthermore, if we denote P'_{opt} and P'_{alg} the respective values of the bounds after doubling, we have

$$\frac{P'_{opt}}{P'_{alg}} = \frac{4P_{opt} - 2(m+r) + p - 2r + 2c_r}{4P_{alg} - 2(m+r) + c_r} \geq \frac{4P_{opt} - 2(m+r) + c_r}{4P_{alg} - 2(m+r) + c_r} \geq \frac{P_{opt}}{P_{alg}},$$

where in the first inequality we have used that $c_r - 2r + p \geq c_r - 2r + s \geq 0$, by **4**, and the second inequality follows from **5**.. So, for the extremum, we can assume there are no linear terms:

$$\frac{P_{opt}}{P_{alg}} \leq \frac{q(m+r) - p^2/2 + pr - r^2}{q(m+r) - rs/2 - (r-s)c_r/2 + t(r-s-t)/2}.$$

Now, using the inequality **5**, we can eliminate m, and obtain

$$\frac{P_{opt}}{P_{alg}} \leq \frac{qc_r - p^2/2 + pr - r^2}{qc_r - rs/2 - (r-s)c_r/2 + t(r-s-t)/2}.$$

The remaining constraints are now **1–4** and **6**. If $s > 0$, and $q > c_r + p - 2s$ we can decrease all variables by one unit, and preserve the above inequalities. In so doing, the value of t does not change, and both the numerator and denominator decrease by the same amount

$$c_r + q - r - 1/2 \geq 0.$$

This implies we can decrease all variables by the same amount until one of two things happen. Either $s = 0$, or $q = c_r + p - 2s$. In the latter case, since we also have that $q \geq c_r$ by **2**, this implies that $2s \leq p$. At this point, after eliminating q (and replacing it by $c_r + p - 2s$), the ratio becomes:

$$\frac{P_{opt}}{P_{alg}} \leq \frac{(c_r + p - 2s)c_r - p^2/2 + pr - r^2}{(c_r + p - 2s)cr - rs/2 - (r - s)c_r/2 + t(r - s - t)/2}.$$

If we decrease all remaining variables by one unit, both the denominator and numerator of the above fraction decrease by

$$c_r + p - r - 2s + 1/2 \geq 0,$$

since $p \geq 2s$ and $c_r \geq r$. And we can continue this process until $s = 0$. In both cases we obtain

$$\frac{P_{opt}}{P_{alg}} \leq \frac{(c_r + p)c_r - p^2/2 + pr - r^2}{(c_r + p)c_r - rc_r/2 + t(r - t)/2}. \tag{12}$$

And we need to maximize this over integral solutions to $r \geq p \geq 0$, $c_r \geq 2r$ and $t = c_r \bmod r$. We consider two cases, depending on the value of c_r.

1. If $c_r = 3r + k$, for $k \geq 0$, and we discard the (nonnegative) term involving t in (12), we obtain:

$$\frac{P_{opt}}{P_{alg}} \leq \frac{8r^2 + 4rp - p^2/2 + k(6r + p + k)}{15r^2/2 + 3rp + k(11r/2 + p + k)}.$$

As an upper bound, we can take the maximum of this value for $k = 0$ and the ratio of the terms involving k, and therefore obtain that:

$$\frac{P_{opt}}{P_{alg}} \leq \max\left(\frac{8r^2 + 4rp - p^2/2}{15r^2/2 + 3rp}, \frac{6r + p + k}{11r/2 + p + k}\right).$$

The second term on this maximum is at most $\frac{12}{11} < \frac{1}{\gamma}$ where γ is our desired bound. The first one, by setting $\alpha = r/p \geq 0$, is equal to

$$\frac{8\alpha^2 + 4\alpha - 1/2}{15\alpha^2/2 + 3\alpha}.$$

This ratio is maximized for $\alpha = \frac{5}{8} + \frac{\sqrt{41}}{8}$, and it achieves a value of

$$\frac{112\sqrt{41} + 656}{99\sqrt{41} + 615} < \frac{1}{\gamma}.$$

2. If $2r \leq c_r < 3r$, then $c_r = 2r + t$, and (12) becomes:

$$\frac{P_{opt}}{P_{alg}} \leq \frac{3r^2 + t^2 - p^2/2 + 3pr + pt + 4rt}{3r^2 + t^2/2 + 2pr + pt + 4rt}.$$

It is easy to see that for any constant C, and fixed values of r and p, the set

$$I = \{t \in [0, r] \ : \ \frac{3r^2 + t^2 - p^2/2 + 3pr + pt + 4rt}{3r^2 + t^2/2 + 2pr + pt + 4rt} \leq C\},$$

is a convex set, and so the maximum value of this ratio is achieved at either $t = 0$ or $t = r$. If we set $t = r$, we obtain

$$\frac{8r^2 + 4pr - p^2/2}{15r^2/2 + 3pr} < \frac{1}{\gamma},$$

```
Algorithm 2: Quickest-Increment for Incremental Valuation
Input   : A valuation function $v$ as above.
Output : A permutation $\sigma$ of $E$
Set $F = \emptyset$;
for $i = 1$ to $r$ do
    Compute $S \subset E \setminus F$ such that $v(F \cup S) \geq v(\emptyset) + i$ and $|S|$ is minimum;
    Set $F = F \cup S$;
end
Output $\sigma$ consistent with how elements were added to S.;
```

as we have already verified. If $t = 0$, we obtain

$$\frac{3r^2 + 3pr - p^2/2}{3r^2 + 2pr},$$

which is maximized at $\alpha = r/p = \frac{1}{2} + \frac{\sqrt{21}}{6}$, with value $\frac{1}{\gamma} = \frac{9}{4} - \frac{\sqrt{21}}{4}$, or $\gamma = \frac{9+\sqrt{21}}{15}$. ◀

This settles the question of how well Quickest-to-Ultimate approximates the maximum incremental matching problem.

5 Upper bound for Quickest-Increment

Quickest-Increment (QI) is another algorithm suggested in [7]. The idea is to increase the size of the current solution by adding as few elements as possible. In that paper, among other results, it was shown that QI has a performance guarantee of 2/3 in the case of bipartite matchings, and also they claim a bound of 3/4 if $r \geq 70$. It is also conjectured there that, as $r \to \infty$, the approximation guarantee for Quickest-Increment approaches 1. We show a family of instances that show that this is false.

Consider the instance H formed by P_7, the path with seven edges, in which the only edges of E_0 are the second and the second to last. Observe that both Q2U and QI have the same performance on this small graph, and it is even optimal. In this small graph we have $q = 5$, $r = 2$ and $m = 2$. There are two incomparable choices for d_i. The first one, given by Q2U, is $d_1 = 2 = d_2$. The second one is given by QI and it is $d_1 = 1$, $d_2 = 4$. They both have value 18, which is optimum.

Now consider the instance G, which is a copies of H. Both algorithms fail to realize the optimum. When considering a copies, we obtain $q = 5a$, $r = 2a$ and $m = 2a$. Algorithm Q2U returns $d_i = 2i$ for $i = 1, \ldots, 2a$, with a value of $f = (5a + 1)(4a) - \sum_{i=1}^{2a} 2i = 16a^2 + 2a$. Algorithm QI return $d_i = i$ for $i = 1, \ldots, a$ and $d_i = 4(i - a) + a$ for $i = a + 1, \ldots, 2a$, with a value of $f = (5a + 1)(4a) - \sum_{i=1}^{a} i - \sum_{i=a+1}^{2a} (4i - 3a) = 33a^2/2 + 3a/2$.

Now suppose we use the QI strategy on k of the a copies and the Q2U strategy on the rest. Then we get $d_i = i$ for $i = 1, \ldots, k$, $d_i = k + 2(i - k)$ for $i = k + 1, \ldots, 2a - k$ and $d_i = 4(i - 2a + k) + 4(a - k) + k$ for $i = 2a - k + 1, \ldots, 2a$, and a value of

$$f = (5a + 1)(4a) - \sum_{i=1}^{k} i - \sum_{i=k+1}^{2a-k} (2(i - k) + k) - \sum_{i=2a-k+1}^{2a} (5(i - 2a + k) + 4(a - k) + k)$$

$$f = 16a^2 + 2a - (3k^2/2 + k/2 - 2ak).$$

Optimizing over k we obtain that for $k = 2a/3$, this solution has a value of $f = 50a^2/3 + 5a/3$. So asymptotically as we take $a \to \infty$ the approximation factor for Q2U approaches $\frac{24}{25}$ and for QI is approaches $\frac{99}{100}$. Note that this family of examples has $r = 2a \to \infty$, and so contradicts the conjecture about QI in [7]. Note this also shows that the approximation guarantee of Q2U is bounded, even when $r \to \infty$. It is possible to show a family of examples that show that when $r \to \infty$, the approximation guarantee for QI approaches $\frac{7}{8}$.

References

1 M. Baxter, T. Elgindy, A. T. Ernst, T. Kalinowski, and M. W. P. Savelsbergh. Incremental network design with shortest paths. *European Journal of Operational Research*, 242:51–62, 2015. doi:10.1016/j.ejor.2014.04.018.

2 V. Chvátal. On certain polytopes associated with graphs. *Journal of Combinatorial Theory, Series B*, 18:138–154, 1975. doi:10.1016/0095-8956(75)90041-6.

3 J. Edmonds. Maximum matching and a polyhedron with 0,1 vertices. *Journal of Research National Bureau of Standards, Section B69*, pages 125–130, 1965. doi:10.6028/jres.069B.013.

4 J. Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965. doi:10.4153/CJM-1965-045-4.

5 J. Edmonds. Submodular functions, matroids, and certain polyhedra. *Combinatorial Structures and Their Applications*, pages 69–87, 1970. doi:10.1007/3-540-36478-1_2.

6 K. Engel, T. Kalinowski, and M. W. P. Savelsbergh. Incremental network design problem with spanning trees. *Journal of Graph Algorithms and Applications*, 2013. arXiv:0902.0885, doi:10.7155/jgaa.00423.

7 T. Kalinowski, D. Matsypura, and M. W. P. Savelsbergh. Incremental network design with maximum flows. *European Journal of Combinatorial Research*, 3:675–684, 2014. doi:10.1016/j.ejor.2014.10.003.

8 G. J. Minty. On maximal independent sets of vertices in claw-free graphs. *Journal of Combinatorial Theory, Series B*, 28:284–304, 1980. doi:10.1016/0095-8956(80)90074-X.

9 S. G. Nurre and T. C. Sharkey. Integrated network design and scheduling problems with parallel identical machines: Complexity results and dispatching rules. *Networks*, 63:306–326, 2014. doi:10.1002/net.21547.

10 N. Sbihi. Algorithme de recherche d'un stable de cardinalité maximum dans un graphe sans étoile. *Discrete Mathematics*, 29:53–76, 1980. doi:10.1016/0012-365X(90)90287-R.

11 A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency.* Springer, 2003.

A Lower bound for integrals of integer convex functions

We prove Lemma 7. Suppose we have a discrete convex function $f : \{0, \ldots, a\} \to \mathbb{N}$, that is, for each $i = 1, \ldots, a-1$, we have

$$f(i+1) - f(i) \geq f(i) - f(i-1).$$

Suppose furthermore that $f(0) = 0$ and define $b = f(a)$. We compute a tight upper bound on the value of $\sum_{i=0}^{a} f(i)$ that depends only on a and b. To this end, define

$$n_k = |\{j \in \{1, \ldots, a\} : f(j) - f(j-1) = k\}|.$$

We must have $n_1 + n_2 + \ldots = a$, and $n_1 + 2n_2 + \ldots = b$, and since f is convex as a sequence, the value of $\sum_{i=0}^{a} f(i)$ in terms of n_k is given by

$$\sum_{i=1}^{n_1} i + \sum_{i=1}^{n_2} (n_1 + 2i) + \sum_{i=1}^{n_3} (n_1 + 2n_2 + 3i) + \ldots = \frac{1}{2}\left(n^T v + n^T A n\right),$$

where n is the vector of the n_k, $v_k = k$, and

$$A = \begin{bmatrix} 1 & 1 & 1 & \dots \\ 1 & 2 & 2 & \dots \\ 1 & 2 & 3 & \dots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix},$$

or $A_{k\ell} = \min(k, \ell)$. This is a symmetric, positive definite matrix, since its Schur complement with respect to entry $(1, 1)$ is just a smaller version of the same matrix, and all its coefficients are positive integers. Note that $n^T v = b$ and so it is a constant independent of the vector of n. The solution given by the following optimization problem gives the required upper bound

$$\begin{array}{ll} \text{maximize} & n^T An \\ \text{subject to} & n_1 + n_2 + \ldots = a \\ & n_1 + 2n_2 + \ldots = b \\ & n_k \in \mathbb{N} \quad k = 1, \ldots, b. \end{array}$$

We will show with the following lemma that a solution n to this problem has at most two consecutive non zeros.

▶ **Lemma 8.** *If there are two positive integers i and j such that $j - i \geq 2$, and $n_i > 0, n_j > 0$, then defining*

$$m = n + (e_{i+1} - e_i) - (e_j - e_{j-1})$$

we have that m is feasible and $m^T Am > n^T An$.

Proof. Note that m is feasible. On the other hand, since A is symmetric

$$m^T Am - n^T An = (m + n)^T A(m - n).$$

Now, $m - n = (e_{i+1} - e_i) - (e_j - e_{j-1})$, and then $A(m - n) = \sum_{k=i+1}^{j-1} e_k$. The coefficients of m and n are nonnegative integers, and furthermore $(m + n)_{i+1} > 0$, which implies the result. ◀

This implies a closed form solution to the problem above.

▶ **Theorem 9.** *Suppose $b = ka + t$, for some integer k, and $t \in \{0, \ldots, a - 1\}$. Then the solution to*

$$\begin{array}{ll} \textit{maximize} & n^T An \\ \textit{subject to} & n_1 + n_2 + \ldots = a \\ & n_1 + 2n_2 + \ldots = b \\ & n_k \in \mathbb{N} \quad k = 1, \ldots, b \end{array}$$

is given by $n_k = (k + 1)a - b = a - t$, $n_{k+1} = b - ka = t$, and its value is $ka^2 + t^2$.

Proof. By the lemma above, the solution has at most two non zeros, and they are adjacent. Let these be ℓ and $\ell + 1$. The solution is given by the solution to $n_\ell + n_{\ell+1} = a$ and $\ell n_\ell + (\ell + 1)n_{\ell+1} = b$. Given that n has two nonzeros, we can compute

$$n^T An = k(a - t)^2 + 2k(a - t)t + (k + 1)t^2 = k(a - t + t)^2 + t^2 = ka^2 + t^2. \quad ◀$$

Lemma 7 is a simple corollary to the above.

Stochastic Unsplittable Flows*

Anupam Gupta[1] and Archit Karandikar[2]

1 **Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA**

2 **Facebook, Inc., Menlo Park, CA, USA**

Abstract

We consider the stochastic unsplittable flow problem: given a graph with edge-capacities, k source-sink pairs with each pair $\{s_j, t_j\}$ having a size S_j and value v_j, the goal is to route the pairs unsplittably while respecting edge capacities to maximize the total value of the routed pairs. However, the size S_j is a random variable and is revealed only after we decide to route pair j. Which pairs should we route, along which paths, and in what order so as to maximize the expected value?

We present results for several cases of the problem under the no-bottleneck assumption. We show a logarithmic approximation algorithm for the single-sink problem on general graphs, considerably improving on the prior results of Chawla and Roughgarden which worked for planar graphs. We present an approximation to the stochastic unsplittable flow problem on directed acyclic graphs, within less than a logarithmic factor of the best known approximation in the non-stochastic setting. We present a non-adaptive strategy on trees that is within a constant factor of the best adaptive strategy, asymptotically matching the best results for the non-stochastic unsplittable flow problem on trees. Finally, we give results for the stochastic unsplittable flow problem on general graphs.

Our techniques include using edge-confluent flows for the single-sink problem in order to control the interaction between flow-paths, and a reduction from general scheduling policies to "safe" ones (i.e., those guaranteeing no capacity violations), which may be of broader interest.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Approximation Algorithms, Stochastic optimization, confluent flows, unsplittable flows

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.7

1 Introduction

We consider the following stochastic problem of routing uncertain demands in a network. We are given a graph $G = (V, E)$ with edge capacities c_e and a set J of k source-sink pairs $\{s_j, t_j\}$ (called *jobs*). We want to route some flow from each source to its corresponding sink, but the amount of flow to be sent for job j (called its *size*) is not known *a priori*. We only know that its size is a random variable S_j, with a known distribution. (We assume that the sizes of jobs are independent of each other.) Each job has a value v_j. We will operate under the prevalent *no-bottleneck assumption* (NBA). In our setting, this means that the maximum size in the support of any job's distribution is at most the minimum capacity of any edge in the graph.

* Research partly supported by NSF awards CCF-1319811, CCF-1540541 and CCF-1617790. Work done when A. Karandikar was at the Computer Science Department, Carnegie Mellon University.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 7; pp. 7:1–7:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We want a routing strategy that decides on jobs to route in the network. This involves us repeatedly choosing an uninstantiated job j and a path P_j for it, and routing this job along the path. Once we do this, the size S_j is instantiated, drawn according to the given probability distribution. If S_j is at most the residual capacity (which initially equals capacity) on each edge of path P_j, the routing is considered successful, we get its value v_j and the residual capacity of all edges in P_j reduces by S_j. Else if there is some edge $e \in P_j$ with residual capacity less than S_j, the routing is unsuccessful and we do not get its value. Moreover, each such "violated" edge is henceforth considered "forbidden" and cannot be used on subsequent paths. When a job j is routed unsuccessfully on a path, it still uses up capacity S_j on all edges along that path that do not become forbidden. The goal is to find a strategy that maximizes the expected cumulative value of jobs it routes successfully. This problem is the stochastic version of the well-known *unsplittable flow problem* (UFP), and as such, we call it the *stochastic unsplittable flow problem* (sUFP).

A strategy for routing jobs is allowed to be *adaptive*, i.e., it can use results of its previous decisions to make its current decision. In contrast *non-adaptive* policies provide a sequence of jobs to route upfront. Storing as well as finding adaptive policies can potentially be exponential in the size of the input and so finding non-adaptive policies of expected value close to the optimal expected value of an adaptive strategy is desirable. The *adaptivity gap* is the ratio of the value of the optimal adaptive strategy to the optimal non-adaptive strategy.

The *stochastic knapsack* problem was first studied by Dean, Goemans and Vondrak [15] who showed that it has a constant adaptivity gap. The stochastic knapsack is the special case of the sUFP on a graph with a single edge. Subsequent work of Dean et al. [14] also considered several versions of the stochastic packing problem, which is a generalization of the sUFP. Over a universe of size d, they showed an $O(\sqrt{d})$ adaptivity gap for stochastic set-packing, and $O(d)$ for general packing problems. Bansal et al. [2] gave an $O(k)$-adaptivity gap for stochastic set-packing with sets of size at most k.

The sUFP was first studied by Chawla and Roughgarden [8]. They studied the *single-sink stochastic routing* problem (SSSR), where all the targets t_i are the same vertex t, and assumed the stronger α-NBA, i.e., the size of each job is supported on $[0, \alpha c_{\min}]$ for some $\alpha < 1$. For planar graphs, they presented a logarithmic approximation algorithm which guaranteed no capacity violations. This work left open several interesting directions: can we work under the NBA instead of the stronger α-NBA? Can we go beyond planar graphs to handle general graphs? How about going beyond single-sink instances, and giving results for more general unsplittable flow instances?

In this paper, we initiate a broader study of the sUFP, and answer these questions in the positive. As is common in the existing research on their deterministic versions, we assume that the underlying graph G is undirected for purposes of the sUFP on trees and general graphs, and that G is a digraph in our treatment of the SSSR and the sUFP on DAGs.

1.1 Our Results

Single-Sink Stochastic Routing Problem (SSSR). Our main result is for the SSSR; as defined above, here all the sinks are co-located. For this result, define the *weight* of job j as $w_j := v_j/\mu_j$, where v_j is the value and $\mu_j = \mathbf{E}[S_j]$ is the expected size of the job.

▶ **Theorem 1.1** (SSSR). *The single-sink stochastic routing problem (under the no-bottleneck assumption) has a poly-time $O(\min(\log k, \log W))$-approximation algorithm. Here k is the number of jobs in the instance, and $W := \frac{\max_j w_j}{\min_j w_j}$ is the maximum ratio between the weights of the jobs.*

Chawla and Roughgarden [8] showed a *safe* $O(\frac{\log W}{1-\alpha})$-approximation for *planar* SSSR instances under the α-NBA; here, safe means the policy is guaranteed to have no edge-capacity violations. No results prior to ours were known for the SSSR on general graphs. In fact, we can also extend Theorem 1.1 to show that under the α-NBA, we get a *safe* $O(\frac{\min(\log k, \log W)}{1-\alpha})$ approximation for the SSSR on general graphs. Recall that for the non-stochastic version of this problem, Dinitz et al. [16] gave a constant-factor approximation algorithm. To obtain the aforementioned logarithmic approximation, we will show a simple general reduction from edge-confluence to node-confluence that was proposed as an open direction by Chen et al. [13].

Stochastic Unsplittable Flow Problem (sUFP) on Directed Acyclic Graphs. Chekuri et al. [10] gave an $O(\sqrt{n})$ approximation for the UFP. We obtain an analogous result in the stochastic setting, giving away further a factor of $O(\sqrt{\log k})$. Recall that k is the number of jobs.

▶ **Theorem 1.2** (sUFP on DAGs). *The stochastic unsplittable flow problem (under the no-bottleneck assumption) has a poly-time $O(\sqrt{n \log k})$-approximation algorithm on directed acyclic graphs.*

Stochastic Unsplittable Flow Problem (sUFP) on Trees. Our next result is for the sUFP on trees. Here, the s_j-t_j paths are unique, which means the routing strategy merely has to decide the sequence of jobs to route.

▶ **Theorem 1.3** (sUFP on Trees). *The stochastic unsplittable flow problem on trees (under the no-bottleneck assumption) has a non-adaptive poly-time $O(1)$-approximation algorithm.*

To the best of our knowledge, this is the first result for the sUFP on trees. Our result follows as a corollary to a more general result, where each job corresponds to a "spider"; we present this result in the full version of this paper. The non-stochastic unsplittable flow problem on trees admits a constant factor approximation under the NBA, by a result of Chekuri et al. [11], and hence our result extends this to the stochastic realm.

Stochastic Unsplittable Flow Problem (sUFP) on General Graphs. For UFP on general graphs, Chakrabarti et al. [7] gave an $O(F_G \log n) = O(\Delta\alpha^{-1} \log^2 n)$ approximation for the UFP. Here F_G denotes the *flow number*, α denotes the expansion and Δ denotes the maximum degree of the graph. (These quantities will be formally defined in Section 4.) Our next result shows how to match these approximation guarantees in the stochastic setting. The proof is contained in the full version of the paper.

▶ **Theorem 1.4** (sUFP). *The stochastic unsplittable flow problem (under the no-bottleneck assumption) has a non-adaptive poly-time $O(d)$-approximation algorithm if the LP relaxation sends flows along paths of length at most d. This can be extended to an $O(F_G \log n) = O(\Delta\alpha^{-1} \log^2 n)$-approximation algorithm on general graphs.*

Safe Routing Strategies. Finally, we give an approach to convert a general strategy (under the NBA) to a safe one (assuming the α-NBA). For the sUFP on general graphs, directed acyclic graphs and trees, we can convert policies under the NBA to safe policies under the α-NBA by sacrificing a factor of $O(\frac{1}{1-\alpha})$ for $\alpha \in (0, \frac{1}{2}]$. For stochastic knapsack and the SSSR such a transformation can be performed for all $\alpha \in (0, 1)$.

1.1.1 Our Techniques

A primary question when dealing with stochastic problems is this: how can we argue about the optimal strategy, which is given by an (exponential-sized) decision-tree? One appealing approach – which we employ here – is to write an "average" LP relaxation which tries to send the average amount of flow for each job. A feasible solution to this linear program is to set the variables for job j based on the probability that the optimal strategy routes j, and hence the LP value gives us an upper bound on the optimal value. However, for stochastic problems, it is not enough to round the solution to integers: indeed, an integer solution to this LP does not directly give us a good strategy (the constraints suffice only when each job behaves like its expectation). Hence we need to "interpret" this solution to get a feasible strategy.

For example, in the SSSR, suppose we are given unsplittable flows that send $\mu_j = \mathbf{E}[S_j]$ amount of flow from s_j to the sink t, for every job j. We may hope to say that each job can be routed with constant probability. However, the flow-paths can interfere in complicated ways, and it is difficult to lower bound the probability that there is "enough room" for some job deep in the process. Our new idea is to alter the flow paths to make them *confluent* – i.e., when two flows use a common edge, they flow together from that point on to the sink. The logarithmic losses come from this step. The confluent flows now behave in a tree-like fashion, and the bottleneck edges are now those incident to the root. We can then argue that these edges are not over-congested with reasonable probability.

For the sUFP in directed acyclic graphs we crucially use our confluence techniques along with idea of v-separation inspired by Chekuri et al for rounding "small" jobs on long flow paths. The other cases are handled using the rounding techniques mentioned above.

To translate arbitrary policies on general graphs to safe policies on unit-capacity graphs, we show how to transform the given set of jobs into new jobs with the same expected size but truncated job sizes, on which we can run the general non-safe strategy. The saved space can then be used to ensure that our real jobs never run out of space.

The sUFP on paths and trees is a natural extension of the well-studied stochastic knapsack problem, and can be viewed as a set of spatially-correlated knapsack problems. Here, we show that for jobs with "large" expected size, we can get good value (comparable to the LP value) by routing an essentially "disjoint" set of jobs. Jobs with small expected size are routed using a scaled-down version of their LP variables. We can then go over the jobs in a certain order, and show that each job, if routed, has a constant probability of having enough capacity to be able to successfully route. A similar plan works for Theorem 1.4 for the sUFP on general graphs: the union bound is over the d edges, and we lose an $O(d)$ term. The translation to the flow number and expansion is standard. The proofs for our results for the sUFP on paths and trees and on general graphs can be found in the full version of this paper.

1.2 Related Work

After the pioneering work of [15], improved algorithms for the stochastic knapsack problem were given by Bhalgat et al. [4, 3], by combining bi-criteria adaptive strategies (another bi-criteria algorithm was given by Li and Yuan [21]), and an LP-rounding approach; we do not know how to implement such adaptive strategies in our case. Work on stochastic knapsack was extended to multi-armed bandits (see, e.g., [17, 18]), and correlated rewards and sizes [19, 21, 22]; all these write more sophisticated LPs to capture correlations. Extending our routing/packing problems to these correlated settings seems non-trivial, and remains an exciting direction for future work. The only prior work on stochastic routing is that of Chawla and Roughgarden [8] discussed above.

Our work on sUFP on trees is directly inspired by work by Chakrabarti et al. [6, 7] and Chekuri et al. [11] on resource-allocation problems and unsplittable flow on paths and trees. These papers get better approximations by combining LP rounding approaches with dynamic programming (DP) for large item sizes, but extending the DP approach to the stochastic case seems difficult. Some variants of the sUFP on Trees (e.g., equal edge-capacities, packing subtrees) are given in the Master's thesis of the second-named author [20]. Algorithms removing the NBA also rely heavily on dynamic programming (see [5, 1] and references), though the LP-based approaches of Chekuri et al. [9] offer hope as well. The unsplittable flow problem, both on general graphs and on trees/paths has been widely studied; see, e.g., the references in [9]. Our results for the sUFP on directed acyclic graphs are based on the work by Chekuri et al [10].

For the single-sink routing problem, we are unable to directly extend the constant-factor approximation of Dinitz et al. [16] to the stochastic case. Instead we use ideas based on confluent flows, which were first developed by Chen et al. [13, 12]. In very recent work, Shepherd, Vetta, and Wilfong [24] showed that for general capacitated networks, under the NBA, there is a $O(\log^6 n)$-approximation algorithm for the demand maximization problem. Shepherd and Vetta [23] give hardness results for such problems.

2 Additional Notation

Here we recall some essential notation introduced in §1 and introduce some new notation. An instance of sUFP consists of a set J containing k jobs, each having a source-sink pair $\{s_j, t_j\}$, a value v_j, and random size S_j. We assume that the distribution of the r.v. S_j is known to us; most of our algorithms only require knowing the mean μ_j. Each edge e of the given graph $G = (V, E)$ has a capacity c_e. Let $c_{\min} := \min_e c_e$ be the minimum capacity of any edge, and $D_{\max} := \min\{d \mid \Pr[S_j > d] = 0 \;\; \forall j \in J\}$. The *no-bottleneck assumption (*NBA*)* requires that $D_{\max} \leq c_{\min}$. By scaling we will always imagine that $c_{\min} = 1$, hence under the NBA, we can assume that $\max_e c_e \leq k$, where k is the total number of jobs. If the sizes are deterministic, we call the problem the *unsplittable flow problem* (UFP); the goal is to route the maximum value set of jobs while respecting edge-capacities. If all sizes are deterministic and equal, we get the *capacitated* EDP (edge-disjoint paths) problem. (In this case we assume that all the jobs are unit-sized, and all the capacities are integers.)

2.1 An LP Relaxation

Given edge capacities $c_e \in \mathbb{R}_{\geq 0}$, and a set J of jobs with demands μ_j, we upper-bound the expected value of the optimal adaptive strategy using the following multicommodity-flow linear program $LP_{UFP}(J, \mathbf{c})$:

$$\begin{aligned} \phi(J, \mathbf{c}) := \qquad & \max \textstyle\sum_j (v_j/\mu_j) x_j & & (LP_{UFP}) \\ & x_j \leq \mu_j & & \forall j \in J \\ & x_j = \textstyle\sum_{P \in \mathcal{P}(s_j,t_j)} f_P & & \forall j \in J \\ & \textstyle\sum_{P: e \in P} f_P \leq c_e & & \forall e \in E \\ & f_P \geq 0 & & \forall P \end{aligned}$$

Here $\mathcal{P}(u, v)$ is the set of all paths from vertex u to v. The same linear program is valid for both directed and undirected instances, with the definition of $\mathcal{P}$ varying between the two. The following theorem is analogous to a result of [15] for stochastic knapsack, and has been used previously [8, 7, 14].

▶ **Theorem 2.1.** *The value of the optimal adaptive strategy for a stochastic routing problem with edge capacity vector* $\mathbf{c}$ *(under the* NBA*, where* $D_{\max} \leq c_{\min} = 1$*) and a set* J *of jobs with expected sizes* $\boldsymbol{\mu}$ *is at most* $\phi(J, \mathbf{c} + \mathbf{1})$*. Using* NBA *and scaling, we get* $\phi(J, \mathbf{c} + \mathbf{1}) \leq \phi(J, 2\mathbf{c}) \leq 2\,\phi(J, \mathbf{c})$*.*

3 Single-Sink Stochastic Routing

We now give a logarithmic approximation for the single-sink stochastic routing (SSSR) problem (under the NBA) on general directed graphs. This improves on the logarithmic guarantee given by Chawla and Roughgarden for planar instances. To understand why this problem is not just the single-sink UFP problem, suppose we are given a routing sending the μ_j flow from each source unsplittably to the sink. To solve the stochastic problem, we have to account for the randomness in the sizes – if we route P_1 and it takes on size greater than its expectation μ_j, what should we do next?

Our main insight is the use of *edge-confluent flows*, which may be somewhat unexpected but is natural in hindsight. A flow in a single-sink network is *confluent* if any two flows which "meet" are merged from there onwards. (One can have edge-confluent or node-confluent flows; the formal definitions appear below.) To get a high-level idea, observe that if we solve the relaxation (LP_{UFP}), and the flow happens to be edge-confluent, the interference between flow-paths can be controlled by controlling the interference on the edges incoming into the sink t.

Our approach is the following: we convert the non-confluent solution to (LP_{UFP}) to an edge-confluent flow. This is not immediate: existing results deal with node-confluence and are applicable only for unit-capacity networks, whereas our SSSR instances have general capacities. Next, we reduce this edge-confluent flow to several instances of the stochastic knapsack problem, one corresponding to every edge incoming to the sink in the unit-capacity network. Our algorithm is adaptive, but only "mildly" so: the adaptivity arises only from the preemption of jobs in each stochastic knapsack instance in order to keep the used-capacity within control. We use the NBA during the conversion to edge-confluent flows. Under the stronger α-NBA, the algorithm is safe.

3.1 Confluent Flows

Given a *directed* graph $G = (V, E)$ with a special sink vertex t, and a set of sources $S = \{s_1, s_2, \ldots, s_k\} \subseteq V$, a *node-confluent flow* is a flow from the sources to the sink such that for each non-sink vertex $v \in V$, all the flow exiting v uses a single arc leaving v. An *edge-confluent* flow is one where for each arc $e \in E$, all flow using this edge must subsequently share the same arcs in their journey to the sink. Equivalently, for an edge-confluent flow $\mathbf{f}$, there exists a mapping $\phi_v : E \to E$ that maps for each vertex v the in-arcs I_v of v to its out-arcs O_v, such that for each out-arc $e = (v, w) \in O_v$, $f_e = \sum_{e' \in I_v : \phi_v(e') = e} f_{e'}$.

For our edge-confluence results, we will operate in a setting where all edges have unit capacity. In this context, we define the the congestion of a flow $\mathbf{f}$ as $\mathsf{cong}(\mathbf{f}) = \max\{1, \max_{e \in E} f_e\}$. We denote the total amount of flow reaching sink t by $|\mathbf{f}|$. The results of Chen et al. [12] for node-confluence can be transferred to edge-confluence to get the following result.

▶ **Theorem 3.1.** *Consider a directed single-sink flow network with* <u>*unit*</u> *edge-capacities under the* NBA*, and a flow* $\mathbf{f}$ *sending* d_i *units of flow from source* s_i *to the sink, respecting edge-capacities. (I.e.,* $\mathsf{cong}(\mathbf{f}) \leq 1$*.) Then the following exist and can be found in polynomial time.*

1. *An edge-confluent flow $\mathbf{f}'$ that for each $i \in [k]$ sends d_i flow from s_i to t (i.e., $|\mathbf{f}'| = |\mathbf{f}|$) so that*

$$\mathsf{cong}(\mathbf{f}') \leq 1 + \ln k \qquad \text{and} \qquad |\mathbf{f}'| = |\mathbf{f}| .$$

2. *A subset $R \subseteq S$ and an edge-confluent flow $\mathbf{f}''$ which for each $i \in R$ sends d_i flow from s_i to t such that $\sum_{i \in R} d_i \geq \frac{1}{3} \sum_{i \in [k]} d_i$, and so that*

$$\mathsf{cong}(\mathbf{f}'') = 1 \qquad \text{and} \qquad |\mathbf{f}''| \geq \frac{|\mathbf{f}|}{3} .$$

Hence, the first result presents a way to route all the jobs while incurring logarithmic congestion, and the second result presents a way to route a large subset of the jobs and incur unit congestion.

Proof Sketch. The idea is to construct the line graph H of the given digraph G (plus one extra node) so that node-confluent flows in the given network correspond to edge-confluent flows in the constructed network. (See Figure 1.) Now given a fractional flow in G, we can map this flow to H, use a result of Chen et al. on transforming general flows to node-confluent flows in H, and transform the resulting node-confluent flow back to an edge-confluent flow in G. The formal proof appears in Appendix A.2. ◀

▶ **Corollary 3.2.** *Consider a directed single-sink flow network with unit edge-capacities under the* NBA*, and a flow $\mathbf{f}$ sending d_i units of flow from source s_i to the sink, respecting edge-capacities. (I.e., $\mathsf{cong}(\mathbf{f}) \leq 1$.) Moreover, each source s_i has weight w_i, and let $\mathbf{w}$ denote the vector of weights.*

Then we can find, in polynomial time, an edge-confluent *flow $\widehat{\mathbf{f}}$ sending $\widehat{d}_i$ units of flow from s_i to the sink respecting edge-capacities (i.e., $\mathsf{cong}(\widehat{\mathbf{f}}) = 1$), such that each $\widehat{d}_i \in [0, d_i]$,*

$$\langle \mathbf{w}, \widehat{\mathbf{d}} \rangle \geq \langle \mathbf{w}, \mathbf{d} \rangle \cdot \frac{1}{\min\{1 + \ln k, 6(1 + \log_2 W)\}},$$

where $W := \frac{\max_j w_j}{\min_j w_j}$.

Proof. We want to find a flow $\widehat{\mathbf{f}}$ for which $\langle \mathbf{w}, \widehat{\mathbf{d}} \rangle$ is within a logarithmic factor of $\langle \mathbf{w}, \mathbf{d} \rangle = \sum_{i=1}^{k} w_i d_i$. Apply Theorem 3.1(1) to the flow $\mathbf{f}$ to obtain edge-confluent flow $\mathbf{f}'$. Scaling the flow $\mathbf{f}'$ down by a factor of $1 + \ln k$ gives us a flow $\widehat{\mathbf{f}}$ with $\langle \mathbf{w}, \widehat{\mathbf{d}} \rangle = \sum_{i \in [k]} w_i \cdot \widehat{d}_i \geq \frac{\langle \mathbf{w}, \mathbf{d} \rangle}{1 + \ln k}$.

Next, bucket the weights w_i into dyadic intervals. By averaging, there exists some interval $I = (2^j, 2^{j+1}]$ such that jobs with weights in this interval have $\sum_{i : w_i \in I} w_i \cdot d_i \geq \frac{\langle \mathbf{w}, \mathbf{d} \rangle}{1 + \log_2 W}$. Use Theorem 3.1(2) to get $R \subseteq \{i : w_i \in I\}$ and flow $\mathbf{f}''$ that sends flow $|\mathbf{f}''| = \sum_{i \in R} d_i \geq \frac{1}{3} \sum_{i : w_i \in I} d_i$. Since the weights of jobs in I are within a factor of 2 of each other, we get that $\sum_{i \in R} w_i \cdot d_i \geq \frac{1}{6} \sum_{i : w_i \in I} w_i \cdot d_i \geq \frac{\langle \mathbf{w}, \mathbf{d} \rangle}{6(1 + \log_2 W)}$. The better of these two edge-confluent flows gives us the claim. ◀

3.2 Approximate Single-Sink Stochastic Routing using Confluent Flows

Consider an instance of the SSSR on the directed graph $G = (V, E)$ under the no-bottleneck assumption (NBA) scaled so that $c_{\min} = 1$. Assume that each source s_i is a unique vertex in G with a single out-edge of capacity 1. This assumption is without loss of generality, since we can always create a new vertex for each source and attach it using a unit-capacity edge to the old location. This does not change feasibility because of the NBA.

For each edge e, define $K_e = \lceil \lfloor c_e \rfloor / 2 \rceil$. Note that $K_e > c_e/3$ for all e. Given such a digraph $G = (V, E)$, define a (multi)graph $G' = (V, E')$ where for each edge $e = (u, v) \in E(G)$, we have K_e parallel unit-capacity edges (u, v) in E'.

The following corollary states a way to obtain a confluent solution to LP_{UFP} for G' within a logarithmic factor of the optimal solution to LP_{UFP} for G. Define the weight w_j for source S_j as v_j/μ_j. Before we state it, recall the definition of LP_{UFP} for G from §2.1 and note that an optimal solution $(\mathbf{x}, \mathbf{f})$ to it is a flow $\mathbf{f}$ in the graph G such that the total weight of this flow $\langle \mathbf{w}, \mathbf{x} \rangle = \sum_{j \in J} w_j x_j = \phi(J, \mathbf{c})$.

▶ **Corollary 3.3.** *Given a solution $(\mathbf{x}, \mathbf{f})$ to LP_{UFP} for the* SSSR *instance on the graph G, there exists an edge-confluent solution $(\mathbf{x}', \mathbf{f}')$ to LP_{UFP} on the unit-capacity graph G', such that*

$$\langle \mathbf{w}, \mathbf{x}' \rangle \geq \langle \mathbf{w}, \mathbf{x} \rangle \cdot \Omega\left(\frac{1}{\min\{\log k, \log W\}} \right),$$

where the weight of job s_j is $w_j = v_j/\mu_j$, and $W := \frac{\max_j w_j}{\min_j w_j}$. Moreover, all the $x'_j \leq \mu_j$ units of flow from s_j to the sink is unsplittably routed. Finally, this flow can be found in time $\mathrm{poly}(n, k)$.

Proof. The solution $(\mathbf{x}, \mathbf{f})$ on G can be ported to a solution $(\widetilde{\mathbf{x}}, \widetilde{\mathbf{f}})$ on G' via scaling down by at most a factor of 3, since the parallel edges replacing each edge e have total capacity $K_e > c_e/3$. Now apply Corollary 3.2 with $d_j = \widetilde{x}_j$, and $w_j = (v_j/\mu_j)$ to get an edge-confluent flow $(\mathbf{x}', \mathbf{f}')$ with the claimed value.

Moreover, since each source has a single out-edge, all flow from s_j to the sink in G' must use this edge. Now edge-confluence ensures that this flow must be routed unsplittably to the sink.

To bound the run-time, observe that the NBA implies that each edge in G need have capacity at most k. Hence G' has at most $n^2 k$ edges. Finally, constructing the graph G' and converting the flow to an unsplittable one can be implemented in polynomial time. ◀

As noted previously, Corollary 3.3 above can be used to obtain an edge-confluent solution $(\mathbf{x}', \mathbf{f}')$ which is within a logarithmic factor of $\phi(J, \mathbf{c})$ where $\mathbf{f}$ is a flow in the graph G'. Let $\partial^-(t)$ denote the set of edges going into the sink in the graph G'. For each such unit-capacity edge $e = \{v, t\} \in \partial^-(t)$, look at the flow over this edge according to $(\mathbf{x}', \mathbf{f}')$. Define E_e as the edges this flow uses on its way from the sources to the sink. By the edge-confluence, if $e \neq e'$ are two edges into the sink, then $E_e \cap E_{e'} = \emptyset$. Moreover, since the flow from each source is routed unsplittably, each job j with $x'_j \neq 0$ has all its flow along edges belonging to a unique E_e. Let J_e be the jobs which are routed along $e \in \partial^-(t)$.

We will now present a strategy for the original SSSR instance on graph G which has expected value at least a constant fraction of $\langle \mathbf{w}, \mathbf{x}' \rangle$. This, along with Theorem 2.1 and Corollary 3.3 will imply that its expected value is within a logarithmic factor of the optimal adaptive strategy for the SSSR instance and prove Theorem 1.1. Note that since the sets $(J_e)_{e \in \partial^-(t)}$ form a partition of $\{j \in J \mid x'_j > 0\}$ we have that

$$\langle \mathbf{w}, \mathbf{x}' \rangle = \sum_{j \in J : x'_j > 0} w_j x'_j = \sum_{e \in \partial^-(t)} \sum_{j \in J_e} w_j x'_j$$

Consider some edge e in $\partial^-(t)$. All flow from sources in J_e flows through e, and hence it is the most congested edge among the ones used by sources in J_e. Indeed, $(\mathbf{x}', \mathbf{f}')$ restricted to these sources gives us a solution to (LP_{UFP}) for a single edge – i.e., for the stochastic knapsack instance $\mathcal{I}_K$ with a unit-capacity knapsack, and a set of jobs J_e.

$$\phi_K(J_e, 1) = \max \left\{ \sum_{j \in J_e} w_j x_j \mid \sum_{j \in J_e} x_j \le 1, \quad x_j \le \mu_j \ \forall j \in J_e \right\}.$$

Having identified these stochastic knapsack instances, let us state the $O(1)$ approximation for the stochastic knapsack provided by Dean, Goemans and Vondrak [15] which we will use to complete the proof.

▶ **Theorem 3.4** (Stochastic Knapsack [15]). *Given an instance of stochastic knapsack with a set of jobs J' there is a non-adaptive strategy $\mathcal{A}_{DGV}$ which gets expected value at least $\frac{7}{16}\,\phi(J', 1) \ge \frac{7}{32}\,OPT$.*

We now run the non-adaptive algorithm in Theorem 3.4 which attains a value $\Omega(1) \cdot \phi_K(J_e, 1)$. For each source $s_i \in J_e$ routed by this algorithm, we route it along the path from s_i to t taken by the confluent flow $\mathbf{f}'$. Once the cumulative size of the routed jobs exceeds 1, we stop routing jobs from J_e. The NBA implies that the jobs in J_e use up a capacity of at most 2.

Note that interference can occur only between instances corresponding to multiple edges $e' \in G'$ which correspond to the same edge e in the original graph G. However, there are only $K_e = \lceil \lfloor c_e \rfloor / 2 \rceil$ such instances and each instance consumes at most 2 units of capacity, the total capacity used is at most $2K_e \le \lfloor c_e \rfloor + 1$, and the set of jobs routable in G' are also routable in G. To see this, note that there is 2 units of space for all but the last of the K_e instances, and for the last instance we still have 1 unit of space which is enough to get full value from each job routed successfully by the stochastic knapsack algorithm in Theorem 3.4. Note that the per-instance preemption of jobs is the reason why our strategy is adaptive, and the availability of less than 2 units of space on the last of the K_e instances is the reason that it is unsafe. This completes the proof of Theorem 1.1.

To obtain safe policies for the SSSR under the stronger assumption of the α-NBA, we first use the approach of Theorem 5.2 to get a safe $O(\frac{1}{1-\alpha})$-approximation for stochastic knapsack under the α-NBA. Now using this for each of the stochastic knapsack instances above gives us a safe approximation for SSSR under the α-NBA. Note that under the stronger α-NBA we can afford to choose $K_e = \lfloor c_e \rfloor$ since we use the safe version of the underlying stochastic knapsack algorithm.

To conclude the section on the SSSR, we note that if we could improve the logarithmic factor in Corollary 3.2 to a constant, then we could use our techniques to get a $O(1)$-approximation for the SSSR problem. This would be implied by a stronger conjecture from Chen et al. [12] that says that given any flow in a network with node congestion 1, one can color the sources using a constant number of colors, such that each chromatic class is node-confluently routable with congestion 1.

4 sUFP on Directed Acyclic Graphs

In this section we give an approximation algorithm for the sUFP on DAGs that extends the work of Chekuri et al. [10, Corollary 1.2] which showed an $O(\sqrt{n})$ approximation for the UFP.

The first idea, as with many UFP results, is to divide the jobs into *small* jobs and *large* jobs. Let us define $\delta = 1/8$. Jobs which have an expected size larger than δ are considered large jobs and the rest are small. Let J_s be the set of small jobs and J_ℓ be the set of large jobs. Observe that $\phi(J_s, \mathbf{c}) + \phi(J_\ell, \mathbf{c}) \geq \phi(J, \mathbf{c})$. Now if we could give, for instances consisting exclusively of small and large jobs, non-adaptive algorithms that obtain at least an $1/\gamma_s$-fraction and $1/\gamma_\ell$-fraction of the respective LP values, then choosing the one with higher guaranteed expected value would give us a non-adaptive strategy obtaining expected value at least an $1/(\gamma_s + \gamma_\ell)$-fraction of the optimal adaptive strategy. (See Fact 1.1.)

For large jobs, we will use the existing result by Chekuri et al. [10, Theorem 1.1]. For small jobs, we use the idea of v-separation from this work [10, Section 3.2] together with our confluence-based techniques to obtain a $O(\sqrt{n}\log k)$ approximation for the sUFP on DAGs (Theorem 1.2). Recall that n is the number of vertices and k is the number of jobs.

4.1 Routing Large Jobs

Let $\mathcal{I} = (G, \mathbf{c}, J)$ be an instance having optimal payoff OPT where all jobs j satisfy $\mu_j \geq \delta$. For these, define the following instance of the UFP on DAGs: let the edge-capacities become $\widehat{c}_e := \lfloor c_e \rfloor$, and we want to find for each job j a unit-sized s_j-t_j path P_j subject to these capacities to maximize the value of the routed paths. The natural LP relaxation for this problem is:

$$\widehat{\phi}(J, \mathbf{c}) := \qquad \max\left\{ \sum_j v_j x_j \mid \sum_{j: e \in P_j} x_j \leq \widehat{c}_e \quad \forall e \in E, \quad \mathbf{x} \in [0,1]^k \right\}. \qquad (LP_{EDP})$$

The theorem in the work by Chekuri et al. [10, Theorem 1.1] implies that we can find, in polynomial-time, a subset $S \subseteq J$ which is feasible for (LP_{EDP}), such that $\sum_{j \in S} v_j \geq \frac{1}{O(\sqrt{n})} \cdot \widehat{\phi}(J, \widehat{\mathbf{c}})$. For the large jobs, assume that all jobs are unit-sized, find the set S, and try to route each of the jobs in S. The NBA implies that the sizes of the jobs are at most 1, and even unit-sized jobs would not violate the edge-capacities. Hence with probability 1 we get a feasible solution to the stochastic UFP on DAGs with value $\frac{1}{O(\sqrt{n})}\,\widehat{\phi}(J, \widehat{\mathbf{c}}) \geq \frac{\delta}{O(\sqrt{n})}\,OPT$. Having shown the approximation result for large jobs, observe that the arguments used above are quite general, and let us record the following theorem for future use.

▶ **Theorem 4.1** (Large Jobs Theorem). *Consider an instance $\mathcal{I} = (G, \mathbf{c}, J_\ell)$ of the* sUFP *(under the* NBA*). Suppose all jobs are δ-large – i.e., they have expected sizes at least $\delta = \delta c_{\min}$. If the integrality gap of the capacitated* EDP *on the graph G is at most γ, then there is a safe*[1] *non-adaptive strategy $\mathcal{A}_\ell$ for* sUFP *that, with probability* 1*, guarantees that value$(\mathcal{A}_\ell) \geq \frac{1}{\gamma} \cdot \widehat{\phi}(J_\ell, \widehat{\mathbf{c}}) \geq \frac{\delta}{4\gamma} \cdot OPT_\ell$.*

4.2 Routing Small Jobs

Let us now examine the case where all jobs are small. The quantity $\phi(J, \mathbf{c})$ represents the weighted flow from the set of sources to the set of sinks. Recall that this quantity is at least half of OPT by Theorem 2.1. We consider all flow paths and partition them into short paths and long paths. For our purposes, paths of length at most $\sqrt{n(1 + \ln k)}$ will be called short paths and the rest will be called long paths. Either the amount of weighted flow along short paths is at least $\phi(J, \mathbf{c})/2$ or that along long paths is at least $\phi(J, \mathbf{c})/2$. We will handle both these cases separately. We will use randomized rounding in both cases.

[1] Note that the strategy described above guarantees that no edge-capacity is violated and is hence safe.

4.2.1 Randomized Rounding for Short Flow Paths

Let $\mathbf{x}$ be the part of the solution to LP_{UFP} which corresponds to the flow along the short paths. If w_i denotes v_i/μ_i then we know that $\sum_{j\in J} w_j x_j$ is at least $\phi(J,\mathbf{c})/2$. We decide to route job j with probability $\alpha x_j/\mu_j$ and job j, if so chosen, is routed on path $P \in \mathcal{P}(s_j,t_j)$ with probability f_P/x_j. With the remaining probability $1-\alpha x_j/\mu_j$ we choose not to route job j. Let Y_{Pj} be the indicator r.v. for whether path P was picked for job j.

We define a random indicator variable Z_{Pj} which indicates if the algorithm $\mathcal{A}$ decides to route job j along path P and there is at least $1/2$ residual capacity on each edge of P at the time of routing it.

$$Z_{Pj} = \begin{cases} 1 & \text{if } Y_{Pj}=1 \text{ and } \sum\limits_{i<j} \sum\limits_{\substack{P'\ni e \\ P'\in\mathcal{P}(s_i,t_i)}} S_i Y_{P'i} \le c_e - 1/2 \text{ for all } e \in P \\ \\ 0 & \text{otherwise} \end{cases} .$$

If we choose $\alpha = \frac{1}{4\sqrt{n(1+\ln k)}}$, we get:

$$\Pr[Z_{Pj}=0 \mid Y_{Pj}=1] \le \sum_{e\in P} \Pr\left[\sum_{i<j} \sum_{\substack{P'\ni e \\ P'\in\mathcal{P}(s_i,t_i)}} S_i Y_{P'i} > c_e/2\right].$$

Note that LP_{UFP} implies

$$\mathbf{E}\left[\sum_{i<j} \sum_{\substack{P'\ni e \\ P'\in\mathcal{P}(s_i,t_i)}} S_i Y_{P'i}\right] = \sum_{i<j} \sum_{\substack{P'\ni e \\ P'\in\mathcal{P}(s_i,t_i)}} \mu_i \cdot (\alpha f_{P'} \ /\mu_i) \le \alpha c_e = \frac{c_e}{4\sqrt{n(1+\ln k)}}.$$

It follows from Markov's inequality that

$$\Pr[Z_{Pj}=0 \mid Y_{Pj}=1] \le \sum_{e\in P} \frac{1}{2\sqrt{n(1+\ln k)}} \le 1/2.$$

We can now complete the proof of the claim, using Markov's inequality once again:

$$\begin{aligned} &\Pr[\text{value obtained from job } j] \\ &\qquad = v_j \cdot \Pr[\text{Job } j \text{ is successfully routed}] \\ &\qquad = v_j \cdot \sum_P \Big(\Pr[Y_{Pj}=1] \cdot \Pr[Z_{Pj}=1 \mid Y_{Pj}=1] \\ &\qquad\qquad \cdot \Pr[\text{Job } j \text{ is successfully routed along } P \mid Z_{Pj}=1, Y_{Pj}=1] \Big) \\ &\qquad \ge v_j \cdot \sum_P \left(\alpha \cdot \frac{f_P}{\mu_j} \cdot \frac{1}{2} \cdot \frac{3}{4}\right) = \frac{3v_j}{8} \cdot \frac{x_j}{\mu_j} \cdot \frac{1}{4\sqrt{n(1+\ln k)}} = \frac{3w_j x_j}{32\sqrt{n(1+\ln k)}}. \end{aligned}$$

4.2.2 Randomized Rounding for Long Flow Paths

Recall that long paths are path of length more than $\sqrt{n(1+\ln k)}$. We examine the case where flow of weight at least $\phi(J,\mathbf{c})/2$ is routed along long paths. As in the proof of SSSR in Section 3, we will convert the flow on the given graph to a flow on a corresponding multigraph with edges of unit capacity. As before, we assume without loss of generality (because of the NBA) that the sources s_j are unique vertices, each with a single out-edge of capacity 1 and similarly, that the targets t_j are unique vertices, each with single in-edge of capacity 1.

As before, we define corresponding to each edge e, an integer $K_e = \lceil \lfloor c_e \rfloor / 2 \rceil$. For the given directed acyclic graph $G = (V, E)$, define a (multi)graph $G' = (V, E')$ where for each edge $e = (u, v) \in E$, we have K_e parallel unit-capacity edges (u, v) in E'. Note that $K_e > c_e/3$ for all e and hence the flow along long paths in G, after being scaled down by a factor of 3 can be converted to a flow in G' preserving flow path lengths. Let this flow, which is of weight at least $\phi(J, \mathbf{c})/6$ correspond to the solution $\mathcal{F} = (\mathbf{x}, \mathbf{f})$ to LP_{UFP} for G'.

All flow paths are long and hence the sum over vertices of the weighted flow routed through each vertex is at least $weight(\mathcal{F}) \cdot \sqrt{n(1 + \ln k)}$. Since the total number of vertices is n, there must exist at least one vertex v through which flow of weight at least $\phi(J, \mathbf{c})/6 \cdot \sqrt{(1 + \ln k)/n}$ is routed. Let $\mathcal{F}_v = (\mathbf{x}_v, \mathbf{f}_v)$ be the solution to LP_{UFP} corresponding to the part of $\mathcal{F}$ routed through v. G' is a directed acyclic multigraph and hence the vertex v splits the flow $\mathcal{F}$ into a single-sink instance G'_{in} and a single-source instance G'_{out}. Let us denote the two corresponding flows by $\mathcal{F}_{v,in}$ and $\mathcal{F}_{v,out}$. We infer[2] from Theorem 3.1 that there exists in G'_{in} an edge-confluent flow $\mathcal{F}'_{v,in}$ of weight $weight(\mathcal{F}_{v,in})/(1 + \ln k)$ and in G'_{out} an edge-confluent flow $\mathcal{F}'_{v,out}$ of weight $weight(\mathcal{F}_{v,out})/(1 + \ln k)$. The flow-per-job and hence the weights of $\mathcal{F}_{v,in}$ and $\mathcal{F}_{v,out}$ are same as the corresponding quantities of $\mathcal{F}_v$. Furthermore these quantities are uniformly scaled down by a factor of $(1 + \ln k)$ in the flows $\mathcal{F}'_{v,in}$ and $\mathcal{F}'_{v,out}$. Hence these two flows can be combined to obtain a flow $\mathcal{F}'_v = (\mathbf{x}'_v, \mathbf{f}'_v)$ of weight $weight(\mathcal{F}_v)/(1 + \ln k) \geq \phi(J, \mathbf{c})/(6\sqrt{n(1 + \ln k)})$ which is confluent in both the partitions G'_{in} and G'_{out}.

We will now devise a routing strategy which has expected value within a constant factor of $weight(\mathcal{F}'_v)$ by randomly rounding this flow. Note that despite the confluence properties of the flow $\mathcal{F}'_v$, we cannot directly reduce this v-separable instance to two instances of the Stochastic Knapsack problem as we did in SSSR, because the routings for both parts must synchronize.

We make an initial decision to route job j with probability $\alpha x'_{vj}/\mu_j$ and job j, if so chosen, is routed on path $P \in \mathcal{P}'(s_j, t_j)$ with probability f'_{vP}/x_j. Here $\mathcal{P}'(s_j, t_j)$ denotes the possible set of paths for job j in graph G'. With the remaining probability $1 - \alpha x'_{vj}/\mu_j$ we choose not to route job j. Let Y_{Pj} be the indicator r.v. for whether path P was picked for job j. Again, we define a random indicator variable Z_{Pj} which indicates if the algorithm $\mathcal{A}$ makes an initial decision to route job j along path P and there is at least $1/2$ residual capacity on each edge of P just before making a decision for job j. If $Z_{Pj} = 0$ even though $Y_{Pj} = 1$ the initial decision is overruled and job is not routed. Otherwise the initial decision holds.

$$Z_{Pj} = \begin{cases} 1 & \text{if } Y_{Pj} = 1 \text{ and } \sum\limits_{i<j} \sum\limits_{\substack{P' \ni e \\ P' \in \mathcal{P}(s_i, t_i)}} S_i Z_{P'i} \leq 1/2 \text{ for all } e \in P \\ 0 & \text{otherwise} \end{cases} .$$

We choose $\alpha = 1/8$. The event $\{Z_{Pj} = 0 \mid Y_{Pj} = 1\}$ occurs if there is at least one edge along P which is congested, i.e., it has residual capacity less than $1/2$ at the time of making a decision for job j. Let e_{in} and e_{out} denote the edges on P which are incoming to and outgoing from v. Since $\mathcal{F}'_v$ is confluent in both G'_{in} and G'_{out}, we infer that if at least one

[2] We scale down by a factor of $(1 + \ln k)$ to ensure unit congestion

edge along P is congested, then either e_{in} or e_{out} must be congested. Hence

$$\Pr[Z_{Pj} = 0 \mid Y_{Pj} = 1] \leq \sum_{e \in \{e_{in}, e_{out}\}} \Pr[\sum_{i<j} \sum_{\substack{P' \ni e \\ P' \in \mathcal{P}(s_i,t_i)}} S_i Y_{P'i} > 1/2].$$

Note that LP_{UFP} implies

$$\mathbf{E}[\sum_{i<j} \sum_{\substack{P' \ni e \\ P' \in \mathcal{P}(s_i,t_i)}} S_i Y_{P'i}] = \sum_{i<j} \sum_{\substack{P' \ni e \\ P' \in \mathcal{P}'(s_i,t_i)}} \mu_i \cdot (\alpha f'_{vP'} / \mu_i) \leq 1/8.$$

Markov's inequality implies that $\Pr[Z_{Pj} = 0 \mid Y_{Pj} = 1] \leq 1/2$. We use Markov's inequality again to infer:

$$\begin{aligned}
&\Pr[\text{value obtained from job } j] \\
&\quad = v_j \cdot \Pr[\text{Job } j \text{ is successfully routed}] \\
&\quad = v_j \cdot \sum_P \Big(\Pr[Y_{Pj} = 1] \cdot \Pr[Z_{Pj} = 1 \mid Y_{Pj} = 1] \\
&\qquad\qquad\qquad \cdot \Pr[\text{Job } j \text{ is successfully routed along } P \mid Z_{Pj} = 1, Y_{Pj} = 1] \Big) \\
&\quad \geq v_j \cdot \sum_P \Big(\alpha \cdot \frac{f_P}{\mu_j} \cdot \frac{1}{2} \cdot \frac{3}{4} \Big) = \frac{3 w_j x_j}{64}.
\end{aligned}$$

Hence the expected value of this rounding strategy is within a constant factor of $weight(\mathcal{F}'_v)$. Finally note that this routing strategy ensures that each edge in G' reaches capacity at most once and hence the capacity consumed on it is at most 2. The definition of G' is such that all but one of the K_e edges in G' will not obstruct each other's jobs from being routed in G if none of the edges ever reaches more than 2 units of congestion and for the last edge the remaining capacity is at least 1, enough to get value from the successfully routed jobs. This completes the proof of Theorem 1.2.

5 Safe Strategies

In this section we address the issue of routing jobs in a way such that we are guaranteed to never overshoot the capacity of any edge. This concept was first studied by Chawla and Roughgarden [8], who called such strategies "safe" strategies. To get non-trivial safe strategies, one has to make an assumption slightly stronger than the NBA. Indeed, we assume that $D_{\max}$, the supremum of the values that any job can take on with non-zero probability, is α where $\alpha \in (0, 1)$ – i.e., the support of each random variable S_j is now $[0, \alpha]$. We refer to this assumption as the α-NBA. As before we have assumed by scaling that $c_{\min} = 1$. Note that we can now get a better upper bound on OPT than what Theorem 2.1 provides : $OPT \leq \phi(J, \mathbf{c} + \alpha \mathbf{1}) \leq (1 + \alpha)\, \phi(J, \mathbf{c})$.

5.1 The Case $\alpha \leq 1/2$

In case $\alpha \in (0, 1/2]$, any strategy for sUFP that is good with respect to the LP relaxation (LP_{UFP}) can be easily converted to a safe strategy with a loss of a factor of $(1 - \alpha)$.

▶ **Theorem 5.1.** *For $\alpha \in (0, 1/2]$, let instance $\mathcal{I} = (G, \mathbf{c}, J)$ of the* sUFP *satisfy the α-*NBA*. Hence, the instance $\mathcal{I}' = (G, \mathbf{c}(1 - \alpha), J)$ satisfies the* NBA*. Given a strategy $\mathcal{A}$ that is an γ-approximation for the instance $\mathcal{I}'$ w.r.t. the LP relaxation (LP_{UFP}), we can obtain a strategy that is an $\frac{\gamma(1+\alpha)}{1-\alpha}$-approximation for $\mathcal{I}$.*

Proof. Observe that $\phi(J, \mathbf{c}(1-\alpha)) \geq \frac{1-\alpha}{1+\alpha} \cdot \phi(J, \mathbf{c}(1+\alpha))$. We know that the strategy $\mathcal{A}$ for $\mathcal{I}'$ achieves expected value at least $\frac{1}{\gamma} \cdot \phi(J, \mathbf{c}(1-\alpha)) \geq \frac{1-\alpha}{\gamma(1+\alpha)} \cdot OPT(\mathcal{I})$. We claim that this run will not violate the actual capacities $\mathbf{c}$ of the edges. Indeed, we can assume, w.l.o.g., that $\mathcal{A}$ does not route any jobs that use any edges that are already forbidden. Hence, just before an edge capacity is violated in the run of $\mathcal{A}$ on $\mathcal{I}'$, it was used to at most $(1-\alpha)c_e$, and the α-NBA ensures that the job can take on size at most $\alpha \leq \alpha\, c_e$. So the total used-up capacity on each edge e is at most $c_e(1-\alpha) + \alpha \leq c_e$, completing the proof. Note that if $\alpha > 1/2$ the instance $\mathcal{I}'$ does not satisfy the NBA. ◀

5.2 The Case $\alpha \geq 1/2$

In this case we give a reduction that takes an arbitrary strategy for sUFP on *unit-capacity* networks which is good with respect to the LP solution, and transforms it into a safe strategy. We use this e.g., on our result for single-sink UFP.

▶ **Theorem 5.2.** *Let $\alpha \in (0,1)$ and $\gamma \geq 1$. Consider a flow network G with unit-capacity edges; i.e., $\mathbf{c} = \mathbf{1}$. (We allow parallel arcs.) Suppose we have strategy $\widetilde{\mathcal{A}}$ that for any instance $\widetilde{\mathcal{I}} = (G, \mathbf{1}, \widetilde{J})$ of the* sUFP *on G satisfying the* NBA*, achieves expected value at least $1/\gamma \cdot \phi(\widetilde{J}, \mathbf{1})$. Then there exists a safe algorithm $\mathcal{A}$ which for all instances $\mathcal{I} = (G, \mathbf{1}, J)$ of the* sUFP *satisfying the α-*NBA *achieves expected value at least $\frac{(1-\alpha)}{6\gamma} \cdot \phi(\widetilde{J}, \mathbf{1}) \geq \frac{(1-\alpha)}{6\gamma(1+\alpha)} \cdot OPT$.*

Proof. First, we can assume that for all jobs $j \in J$, we have $\Pr[S_j = 0] = 0$, by losing a factor of 2 in the approximation. To prove this, first imagine there are no jobs having mean size zero, since these can be routed without using any capacity. Let $\mu_{\min} := \min_{j \in J} \mu_j$ be the least mean job size. We transform the jobs in J to be supported on $[\mu_{\min}, \alpha]$ by defining their new size to be $S'_j := \max\{\mu_{\min}, S_j\}$. This increases the mean μ_j of each job j to $\mu_{j'} \leq \mu_j + \mu_{\min} \leq 2\mu_j$, but still satisfies the α-NBA. Consequently, the value of the LP has decreased by at most 2 and our strict positivity assumption is justified. Any strategy safe for the modified instance is also safe for the original instance. An advantage of having strictly positive job sizes is that we can argue that if the given strategy $\widetilde{\mathcal{A}}$ routes a job on some path P, all edges on the path have strictly positive residual capacity (and are not forbidden, of course). If not, if there were some edge of capacity zero, or a forbidden edge, the routing would necessarily be unsuccessful, and we could drop it without any loss in value.

Now define $\alpha' := 1 - \alpha$ and $\delta := \alpha'/2$. Separate the jobs into δ-large (those with $\mu_j \geq \delta$) and δ-small (the remaining). For the δ-large jobs, apply Theorem 4.1 to obtain a safe $(1+\alpha)\gamma/\delta$-approximation. Observe that to apply Theorem 4.1, we need an algorithm for the capacitated UFP problem – however, our assumed algorithm $\widetilde{\mathcal{A}}$ for sUFP is at least as powerful, and hence suffices. (The approximation factor is better than claimed in Theorem 4.1, since (i) we start with unit edge-capacities, and hence we do not need to round down the capacities (ii) We use the better $(1+\alpha)\phi(J, \mathbf{c})$ upper bound on the OPT)

For the δ-small jobs J_s, let us denote the original small instance by $\mathcal{I} = (G, \mathbf{1}, J_s)$, and define a modified instance $\mathcal{I}' = (G, \alpha'\mathbf{1}, J'_s)$. For each job $j \in J_s$, find a threshold $\ell_j \leq \mu_j$ such that

$$\mathbf{E}[\max(\ell_j, \min(S_j, \alpha'))] = \mu_j.$$

In words, we "clip" the job size S_j at α' on the upper side, and at ℓ_j on the lower side, and want the mean to remain unchanged. This is possible since $\mu_j \leq \delta < \alpha'$, so the upper clipping brings the mean down, which the lower clipping can remedy. Now define the size of the new job j' to be $S_{j'} := \max(\ell_j, \min(S_j, \alpha'))$, this clipped random variable, and let $\mu_{j'} := \mathbf{E}[S_{j'}]$.

Observe that $S_j, S_{j'}$ are coupled by definition, such that conditioned on $S_{j'} < \alpha'$, we know that $S_{j'} \geq S_j$.

Observe that the value of the LP relaxation $\phi(J'_s, \alpha'\mathbf{1}) \geq \alpha' \cdot \phi(J_s, \mathbf{1})$. Moreover, the instance $\mathcal{I}' = (G, \alpha'\mathbf{1}, J'_s)$ satisfies the NBA, and hence running $\widetilde{\mathcal{A}}$ on J'_s achieves an expected value of at least $1/\gamma \cdot \phi(J'_s, \alpha'\mathbf{1})$. So we can imagine executing the algorithm $\widetilde{\mathcal{A}}$ on $\mathcal{I}'$, whilst actually routing the jobs in $\mathcal{I}$. I.e., when $\widetilde{\mathcal{A}}$ asks to route job j', we actually run job j, it takes on size S_j, and we report back the size $S_{j'}$ to the algorithm $\widetilde{\mathcal{A}}$. As argued above, we assume that $\widetilde{\mathcal{A}}$ does not route any job on forbidden edges, or edges of zero capacity.

The crucial observation is that conditioned on an edge e's capacity having been used up to less than α', the actual usage (according to the real job sizes S_j) is no more than the usage according to the job sizes $S_{j'}$ reported to $\widetilde{\mathcal{A}}$. This is because, conditioned on $S_{j'} < \alpha'$, we know that $S_{j'} \geq S_j$.

Now to see that this strategy is safe for $\mathcal{I}$, consider an edge on some path P on which $\widetilde{\mathcal{A}}$ routes some job j. Previously e's capacity was used up to less than α' (since it had non-zero residual capacity by our assumption on $\widetilde{\mathcal{A}}$), and even if the current job uses it to its maximum size $\alpha = 1 - \alpha'$, we will not violate the actual capacity. Hence, any job that is routed in $\widetilde{\mathcal{A}}$'s run on $\mathcal{I}'$ will also be successful routed in the run on $\mathcal{I}$.

This gives us an $(1+\alpha)\gamma/\alpha'$-approximation algorithm for small jobs. Using Fact 1.1 the better of the two gives us a $\frac{3\gamma(1+\alpha)}{(1-\alpha)}$-approximation. Moreover, losing another factor of 2 for the transformation to strictly positive job sizes gives us the result. ◀

Theorem 5.2 is the only result in our paper where we require knowing more information about the distribution of S_j beyond just the expectation μ_j. Now we can use Theorems 5.1 and 5.2 to give a safe strategy for variants of the sUFP. In particular, applying this to the stochastic knapsack result from Theorem 3.4 gives us a safe algorithm for that problem, and hence for the SSSR.

6 Conclusions and Discussion

In this paper we gave approximation algorithms for stochastic routing problems under the no-bottleneck assumption. These problems generalize the classical unsplittable flow problem. Our results include improved results for the single-sink case, constant-factor approximations for stochastic routing on trees and paths, and results for general graphs as well. We also gave techniques to convert unsafe strategies into safe ones, for unit capacity networks. Many interesting open questions remain: E.g., can we get a constant-factor for the single-sink setting? Can we give results without the no-bottleneck assumption?

References

1 Aris Anagnostopoulos, Fabrizio Grandoni, Stefano Leonardi, and Andreas Wiese. A mazing $2+\epsilon$ approximation for unsplittable flow on a path. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 26–41, 2014. doi:10.1137/1.9781611973402.3.

2 Nikhil Bansal, Anupam Gupta, Jian Li, Julián Mestre, Viswanath Nagarajan, and Atri Rudra. When LP is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica*, 63(4):733–762, 2012.

3 Anand Bhalgat. A $(2+\epsilon)$-approximation algorithm for the stochastic knapsack problem. At http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.222.7341&rep=rep1&type=pdf, 2011.

4 Anand Bhalgat, Ashish Goel, and Sanjeev Khanna. Improved approximation results for stochastic knapsack problems. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1647–1665, 2011.

5 Paul S. Bonsma, Jens Schulz, and Andreas Wiese. A constant factor approximation algorithm for unsplittable flow on paths. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 47–56, 2011. doi:10.1109/FOCS.2011.10.

6 Gruia Calinescu, Amit Chakrabarti, Howard Karloff, and Yuval Rabani. An improved approximation algorithm for resource allocation. *ACM Trans. Algorithms*, 7(4):Art. 48, 7, 2011. doi:10.1145/2000807.2000816.

7 Amit Chakrabarti, Chandra Chekuri, Anupam Gupta, and Amit Kumar. Approximation algorithms for the unsplittable flow problem. *Algorithmica*, 47(1):53–78, 2007. doi:10.1007/s00453-006-1210-5.

8 Shuchi Chawla and Tim Roughgarden. Single-source stochastic routing. In *Proceedings of APPROX*, pages 82–94. Springer, 2006.

9 Chandra Chekuri, Alina Ene, and Nitish Korula. Unsplittable flow in paths and trees and column-restricted packing integer programs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings*, pages 42–55, 2009. doi:10.1007/978-3-642-03685-9_4.

10 Chandra Chekuri, Sanjeev Khanna, and F. Bruce Shepherd. An o(sqrt(n)) approximation and integrality gap for disjoint paths and unsplittable flow. *Theory of Computing*, 2(7):137–146, 2006. doi:10.4086/toc.2006.v002a007.

11 Chandra Chekuri, Marcelo Mydlarz, and F. Bruce Shepherd. Multicommodity demand flow in a tree and packing integer programs. *ACM Trans. Algorithms*, 3(3):Art. 27, 23, 2007. doi:10.1145/1273340.1273343.

12 Jiangzhuo Chen, Robert D. Kleinberg, László Lovász, Rajmohan Rajaraman, Ravi Sundaram, and Adrian Vetta. (Almost) tight bounds and existence theorems for single-commodity confluent flows. *J. ACM*, 54(4):Art. 16, 32 pp. (electronic), 2007. doi:10.1145/1255443.1255444.

13 Jiangzhuo Chen, Rajmohan Rajaraman, and Ravi Sundaram. Meet and merge: approximation algorithms for confluent flows. *J. Comput. System Sci.*, 72(3):468–489, 2006. doi:10.1016/j.jcss.2005.09.009.

14 Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Adaptivity and approximation for stochastic packing problems. In *SODA*, pages 395–404, 2005.

15 Brian C. Dean, Michel X. Goemans, and Jan Vondrák. Approximating the stochastic knapsack problem: The benefit of adaptivity. *Math. Oper. Res.*, 33(4):945–964, 2008. doi:10.1287/moor.1080.0330.

16 Yefim Dinitz, Naveen Garg, and Michel X. Goemans. On the single-source unsplittable flow problem. *Combinatorica*, 19(1):17–41, 1999. doi:10.1007/s004930050043.

17 Sudipto Guha and Kamesh Munagala. Approximation algorithms for budgeted learning problems. In *ACM Symposium on Theory of Computing (STOC)*, pages 104–113. ACM, 2007. Full version as *Sequential Design of Experiments via Linear Programming*, http://arxiv.org/abs/0805.2630v1.

18 Sudipto Guha and Kamesh Munagala. Approximation algorithms for bayesian multi-armed bandit problems. *CoRR*, abs/1306.3525, 2013. URL: http://arxiv.org/abs/1306.3525.

19 Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, and R. Ravi. Approximation algorithms for correlated knapsacks and non-martingale bandits. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 827–836, 2011.

20 Archit Karandikar. Approximation algorithms for stochastic unsplittable flow problems. Master's thesis, Carnegie Mellon University, 2015. URL: https://github.com/architkarandikar/MastersThesis.

21 Jian Li and Wen Yuan. Stochastic combinatorial optimization via poisson approximation. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 971–980, 2013. doi:10.1145/2488608.2488731.

22 Will Ma. Improvements and generalizations of stochastic knapsack and multi-armed bandit approximation algorithms: Extended abstract. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1154–1163, 2014. doi:10.1137/1.9781611973402.85.

23 F. Bruce Shepherd and Adrian Vetta. The inapproximability of maximum single-sink unsplittable, priority and confluent flow problems. *CoRR*, abs/1504.00627, 2015. URL: http://arxiv.org/abs/1504.00627.

24 F. Bruce Shepherd, Adrian Vetta, and Gordon T. Wilfong. Polylogarithmic approximations for the capacitated single-sink confluent flow problem. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 748–758, 2015. doi:10.1109/FOCS.2015.51.

A Missing Proofs

A.1 Combining Results for Small and Large Jobs

▶ **Fact 1.1** (Combination). *Consider an instance $\mathcal{I} = (G, \mathbf{c}, J)$ of the* sUFP *with optimal payoff OPT specified by the graph G, edge-capacity vector $\mathbf{c}$ and the set of jobs J. Let J_1 and J_2 form a partition of J and consider instances $\mathcal{I}_1 = (G, \mathbf{c}, J_1)$ and $\mathcal{I}_2 = (G, \mathbf{c}, J_2)$ with optimal payoffs OPT_1 and OPT_2. Suppose for each instance $\mathcal{I}_i$, there exists a polytime non-adaptive algorithm $\mathcal{A}_i$, a polytime computable quantity ξ_i and a quantity $\gamma_i \geq 1$ such that $\mathbf{E}[payoff(\mathcal{A}_i)] \geq \xi_i \geq \frac{1}{\gamma_i} OPT_i$. Then the algorithm $\mathcal{A}$ that returns the solution for the instance $\mathcal{I}_i$ with the higher ξ_i has*

$$\mathbf{E}[payoff(\mathcal{A})] \geq \frac{1}{\gamma_1 + \gamma_2} OPT\,.$$

Proof. Since J_1 and J_2 form a partition of J, $OPT \leq OPT_1 + OPT_2 \leq \gamma_1 \xi_1 + \gamma_2 \xi_2$. Hence $\max(\xi_1, \xi_2) \geq \frac{OPT}{\gamma_1 + \gamma_2}$.

As an example application, suppose we have two different LP rounding algorithms that on instances J_i produce solutions with values $\xi_i = \frac{1}{\gamma_i} OPT_i$. Then taking the larger one is an $(\gamma_1 + \gamma_2)$-approximation. ◀

A.2 Reducing Edge-Confluence to Node-Confluence

The results of Chen et al. [12], along with most other literature address node-confluence. We show how to get Theorem 3.1 on edge-confluence from these resuts. The existing result of Chen et al. that we use is the analog of Theorem 3.1 for node-confluent flows under node-congestion. The node-congestion of a flow is defined as $\mathsf{n\text{-}cong}(\mathbf{f}) = \max\{1, \max_{v \in V \setminus \{t\}} f_v\}$ where f_v denotes the flow passing though vertex v.

Proof. Theorem 3.1 Consider any digraph $G = (V, E)$ with sources $S_G \subseteq V$, sink node t, and unit edge-capacities, in which we have a general flow $\mathbf{f}$ respecting edge capacities(i.e. $\mathsf{cong}(\mathbf{f}) = 1$) that we wish to convert to an edge-confluent flow. We assume w.l.o.g. that this flow is acyclic and that each source has exactly one outgoing edge from it and no edges incoming

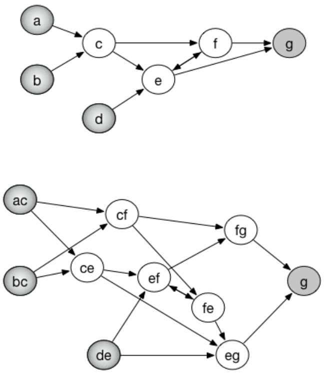


Figure 1 Reducing edge-confluence to node-confluence. Edge-confluent flows in the network above correspond to node-confluent flows in the one below. The sink is shaded grey. The sources are shaded with a gradient.

into it. To justify the latter, note we can augment the graph by adding a new source for each orginal source, connected to it by a unit-capacity edge. Under the NBA, this transformation does not change congestion and flows which are edge-confluent in the augmented graph are also edge-confluent in the original graph.

We construct another graph H with unit node-capacities, which is essentially the directed line graph of G (plus one extra node). This construction is demonstrated in Figure 1. The graph H has a node v_{pq} for every arc (p, q) in G. There is an arc from v_{pq} to v_{rs} exactly when $q = r$. Moreover, it has node v_t, with arcs from all nodes v_{pt} to v_t. Finally, for every source s there is exactly one graph (s, x_s) leaving s as per our assumption. The set of sources in the new graph H is defined by $S_H = \{v_{sx_s} \mid s \in S\}$.

Now given the flow $\mathbf{f}$ in G from sources S to sink t, take any path decomposition of the flow $\mathbf{f}$. Each flow path P can be mapped in a natural way to a flow-path in H: if $P = \langle s, a, b, \dots, z, t\rangle$, then it is mapped to path $\langle v_{sa}, v_{ab}, \dots, v_{zt}, v_t\rangle$ in H. Doing this for all flow-paths gives a flow $\mathbf{h}$ in H. The node-capacities in H are satisfied by $\mathbf{h}$ because the edge-capacities in G were satisfied by $\mathbf{f}$. This is a injection from unit edge-congestion flows in G into unit node-congestion flows in H. Note that the procedure can be reversed to obtain a surjection from unit node-congestion flows in H onto unit edge-congestion flows in G. These mappings are not inverses since several flows in H may correspond to a single flow in G[3]. Under these mappings, any edge-confluent flow $\mathbf{f}$ from S_G to t in G is mapped to a node-confluent flow $\mathbf{h}$ from S_H to v_t in H, and vice versa. Since the flow though an edge in G equals flow through the corresponding vertex in H we note that $\mathsf{cong}(\mathbf{f}) = \mathsf{n\text{-}cong}(\mathbf{h})$.

Hence, to prove Theorem 3.1, we do the following: we take the unit-congestion flow $\mathbf{f}$ in G and convert it into a unit-node-congestion flow $\mathbf{h}$ in H. Now we can use results of Chen et al. [12] on node-confluent flows in unit-capacity graphs. They show how to convert $\mathbf{h}$ into:

- A node-confluent flow $\mathbf{h}'$ with $\mathsf{n\text{-}cong}(\mathbf{h}') = 1 + \ln k$.
- A node-confluent flow $\mathbf{h}''$ respecting node-capacities (i.e. $\mathsf{n\text{-}cong}(\mathbf{h}'') = 1$) that routes flow from a subset of the sources in S_H having total flow at least a third of the total original flow in $\mathbf{h}$.

Mapping these flows back to G gives us the flows claimed in Theorem 3.1. ◀

[3] Consider node c in Figure 1 and note that $(a \to c \to e, b \to c \to f)$ and $(a \to c \to f, b \to c \to e)$ can be two different alternatives for path decomposition.

A reduction from node-confluence to edge-confluence is easy as Chen et al. [13] had previously observed. This result thus identifies an interconvertability between node-confluence and edge-confluence. Note that we have used this construction for unit capacity networks only since it suffices our purpose of addressing the SSSR. However this argument also extends in a straightforward way to capacitated graphs. Hence it can also be used to obtain a $O(\log^6 n)$ approximate edge-confluent flow of congestion 2 corresponding to the recent node-confluence results of Shepherd, Vetta, and Wilfong [24, Theorem I.7].

Streaming Complexity of Approximating Max 2CSP and Max Acyclic Subgraph

Venkatesan Guruswami*[1], Ameya Velingker†[2], and Santhoshini Velusamy‡[3]

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

2 School of Computer and Communication Sciences, EPFL, Lausanne, Switzerland
ameya.velingker@epfl.ch

3 Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India
cs13b059@smail.iitm.ac.in

Abstract

We study the complexity of estimating the optimum value of a Boolean 2CSP (arity two constraint satisfaction problem) in the single-pass streaming setting, where the algorithm is presented the constraints in an arbitrary order. We give a streaming algorithm to estimate the optimum within a factor approaching 2/5 using logarithmic space, with high probability. This beats the trivial factor 1/4 estimate obtained by simply outputting 1/4th of the total number of constraints.

The inspiration for our work is a lower bound of Kapralov, Khanna, and Sudan (SODA '15) who showed that a similar trivial estimate (of factor 1/2) is the best one can do for Max CUT. This lower bound implies that beating a factor 1/2 for Max DICUT (a special case of Max 2CSP), in particular, to distinguish between the case when the optimum is $m/2$ versus when it is at most $(1/4+\epsilon)m$, where m is the total number of edges, requires polynomial space. We complement this hardness result by showing that for DICUT, one can distinguish between the case in which the optimum exceeds $(1/2+\epsilon)m$ and the case in which it is close to $m/4$.

We also prove that estimating the size of the maximum acyclic subgraph of a directed graph, when its edges are presented in a single-pass stream, within a factor better than 7/8 requires polynomial space.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.1.6 Optimization, G.2.1 Combinatorics, G.2.2 Graph Theory

Keywords and phrases approximation algorithms, constraint satisfaction problems, optimization, hardness of approximation, maximum acyclic subgraph

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.8

1 Introduction

We are concerned with the ability of single-pass streaming algorithms to estimate the optimum value of constraint satisfaction problems (CSPs), focusing in particular on very

* Research supported in part by NSF grant CCF-1526092.
† Work done mostly while at CMU.
‡ Work done mostly while at CMU, Pittsburgh during S.N. Bose Scholars Program 2016 conducted by the Science and Engineering Board (SERB), Department of Science and Technology (DST), Govt. of India, the Indo-U.S. Science and Technology Forum (IUSSTF) and WINStep Forward.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 8; pp. 8:1–8:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

simple (Boolean, arity two) constraints. The impetus for our investigation is a striking lower bound result by Khanna, Kapralov, Sudan [16] for the problem of estimating the Max Cut in a graph, when the edges arrive one-by-one in a streaming fashion. There is a trivial factor 1/2-approximation for the problem using only $O(\log n)$ space, namely, count the number of edges and output half this value as the estimate for Max Cut value.[1] The authors of [16] showed that even with $\tilde{\Omega}(\sqrt{n})$ space, a single-pass streaming algorithm cannot achieve a factor $(1/2+\epsilon)$-approximation, for any constant $\epsilon > 0$.[2] The lower bound in fact holds even if the edges arrive in a random (as opposed to worst-case) order. A later work shows that obtaining a β-approximation, for some β bounded away 1 requires $\Omega(n)$ space [17]. In contrast, there are streaming algorithms producing a $(1-\epsilon)$-approximation in $\tilde{O}_\epsilon(n)$ space, by use of "cut sparsifiers" [2, 19].

1.1 Context: Approximation resistance of CSPs

The Max Cut problem is a particular, most basic form of CSP, with underlying constraints being of the form $x \neq y$. More generally, a CSP over domain D is specified by a template $\Lambda = \{P_1, \ldots, P_s\}$ of predicates $P_i : D^{a_i} \to \{0,1\}$ (a_i is the arity of P_i), and an instance of MaxCSP(Λ) is specified by a variable set V and a collection of "constraint tuples" (i, τ) with $i \in \{1, 2, \ldots, s\}$ denoting the type of constraint and $\tau \in V^{a_i}$ denoting the tuple of variables to which the constraint is applied. The goal is to find an assignment $\sigma : V \to D$ so that a maximum number of constraints are satisfied, where a constraint (i, τ) is satisfied by σ if $P_i(\sigma(\tau_1), \ldots, \sigma(\tau_{a_i})) = 1$ (in other words, setting the variables in the scope of this constraint according to assignment σ satisfies the predicate P_i). The maximum possible number of satisfied constraints is called the optimum value of the CSP instance. For most templates Λ, the Max CSP problem is NP-hard to solve optimally (see [20] for a dichotomy theorem for Max CSP classifying the rare easy cases). So there has been a lot of work on designing approximation algorithms. An absolutely trivial algorithm is the random assignment algorithm that ignores the instance structure, and simply assigns a random value to each variable. This achieves a α_Λ approximation for MaxCSP(Λ) where $\alpha_\Lambda = \min_i\{\mathbb{E}[P_i]\}$, with the expectation taken over a random input to P_i – we call α_Λ the *random assignment threshold*. Since the seminal work of Håstad [13] it has been established that for several interesting CSPs, it is NP-hard to beat the performance ratio of this trivial algorithm! Such CSPs are called approximation resistant in the literature (see, for instance, [8] and references therein). Already for arity 3, several important CSPs such as Max E3SAT and Max E3LIN (linear equations mod 2) are approximation resistant.

Thanks to semidefinite programming, for arity 2 CSPs, one can do better than the α_Λ factor [7, 14]. In particular, for (Boolean) Max 2CSP, where the domain is $D = \{0,1\}$ and Λ includes all predicates of arity 2, the seminal work of Goemans and Williamson [7], gave a factor 0.79607 algorithm (this ratio was further improved to 0.8593 in [4]).[3] The GW algorithm was a substantial improvement over the random assignment threshold of 1/4 which was also the best known algorithm for Max 2CSP at that time. For the specific case of Max Cut, Goemans and Williamson get the famous 0.87856 approximation factor, a vast

[1] We use numbers < 1 to designate the approximation ratio for the maximization problems we study: a factor γ approximation means the output estimate is at least γ times the optimum, and always at most the optimum.

[2] Throughout, we allow streaming algorithms to be randomized, and their estimate should satisfy the approximation guarantee with probability say 9/10.

[3] This guarantee is stated for the Max DICUT problem, which is a CSP with a single predicate $P(x, y) = \overline{x} \wedge y$, but in fact it holds for Max 2CSP in general.

improvement over the random assignment threshold of 1/2 (which was again the best known algorithm at that point).

The aforementioned Khanna, Kapralov, Sudan result [16], however, shows that in the streaming model, Max Cut is approximation resistant! Thus, streaming algorithms cannot non-trivially estimate the optimum of even the simplest CSP. This raises the question whether streaming algorithms operating in small space can non-trivially approximate (i.e., beat the random assignment threshold) for *some* CSP, or whether every CSP is approximation resistant in the streaming model.

1.2 Our results for Max 2CSP and Max DICUT

In this work, we give a factor 2/5 streaming algorithm for Max 2CSP that uses $O(\log n)$ space. In particular, this beats the random assignment threshold of 1/4.

▶ **Theorem 1.** *Fix any $\gamma > 0$. There is an efficient single-pass streaming algorithm that, given as input a Max 2CSP instance on n variables, with constraints arriving one-by-one in an arbitrary order, uses $O_\gamma(\log n)$ space and with probability at least $9/10$ outputs an estimate in the range $[(2/5 - \gamma)OPT, OPT]$, where OPT is the optimum value of the CSP instance.*

Any arity 2 Boolean predicate can be expressed as the disjunction of (at most 4) AND constraints, at most one of which can be satisfied by any assignment. By AND constraints, we mean one of the predicates $(x \wedge y)$, $(\overline{x} \wedge y)$, $(x \wedge \overline{y})$, and $(\overline{x} \wedge \overline{y})$ (that is, we take an AND of two *literals*). Any Max 2CSP instance can thus be mapped into a Max 2AND instance with the same optimum value. The above theorem therefore follows from our result about Max 2AND stated below (without loss of generality, in the rest of the paper, we only focus on Max 2AND and not Max 2CSP):

▶ **Theorem 2.** *Fix any $\gamma > 0$. There is an $O_\gamma(\log n)$ space single-pass streaming algorithm that can estimate the optimum value of a Max 2AND instance, whose AND constraints arrive in an arbitrary order, within a factor of $2/5 - \gamma$ with probability at least $9/10$. More specifically, on an instance with m constraints and optimum value OPT, the algorithm outputs a lower estimate on OPT which, with probability at least $9/10$, lies in the range $[(2/5 - \gamma)OPT, OPT]$.*

Our algorithm and analysis are simple and elementary, and are based on the combination of two observations. The first is that the bias of instance, which is sum over all variables of $|\text{pos}_v - \text{neg}_v|$ where pos_v (resp. neg_v) is the number of AND constraints in which v participates as a positive literal (resp. negated literal), is a good proxy for the optimum value when the optimum is large. The second is that the bias can be estimated efficiently in a streaming fashion via L_1 norm estimation of a vector under bounded dynamic updates of its coordinates.

Note that prior to 1994 there was *no* efficient algorithm known to approximate Max 2AND (or even the restricted Max DICUT problem) within a factor better than the random assignment threshold of 1/4. So, its simplicity notwithstanding, it is perhaps surprising that one can in fact have a low-space and time-efficient streaming algorithm that achieves a factor much better then 1/4.

Since the semidefinite programming based approximation for Max DICUT, many works have also given simpler algorithms that beat the factor 1/4. We mention some of them here. Trevisan used randomized rounding of a natural linear program to give a factor 1/2 algorithm [21]. Alimonti obtained a factor 1/3 approximation using local search [1]. Halperin and Zwick presented simple factor 2/5 and 9/20 algorithms based on some path removal ideas, and also a factor 1/2 algorithm (via a combinatorial method to find a half-integral LP solution) [11]. (In Appendix A, we give a different proof of the half-integrality of the LP, and the associated (non-streaming) factor 1/2 algorithm.) Feige and Jozeph [5] give a very simple factor 2/5 algorithm for Max DICUT: take the greedy cut which sets variables whose out-degree is at least their in-degree to 0, and remaining to 1, and return the better of this cut and a uniformly random cut.

None of these algorithms seem to have an efficient streaming implementation. The closest to our algorithm is the greedy algorithm in [5], and we are able to get a streaming friendly estimate of the DICUT value by avoiding computation of the greedy cut, but instead the total bias of all vertices. Further our approach extends naturally to Max 2AND.

Hardness of factor $1/2 + \epsilon$ approximation. We do not know if the 2/5 approximation factor for Max 2AND is the best possible in the streaming model. However, one cannot achieve an approximation factor larger than 1/2. This is because, by a trivial reduction from the streaming lower bound for Max Cut in [16], we can deduce the following hardness even for the special case of Max DICUT.

▶ **Theorem 3.** *For any constant $\epsilon > 0$, a factor $(1/2+\epsilon)$ randomized streaming approximation algorithm for Max DICUT must use space $\tilde{\Omega}(\sqrt{n})$. Specifically, a randomized streaming algorithm that can decide, with success probability $9/10$, whether an m edge directed graph has a dicut of value at least $m/2$ or has no dicut of value $(1/4+\epsilon)m$, requires $\tilde{\Omega}(\sqrt{n})$ space.*

Complementary algorithmic result. We show the tightness of the above hardness result via the following algorithmic result for Max DICUT. The approach is again based on estimation of the bias of the graph: we prove that graphs whose dicut value is close to $m/4$ must have small bias, and graphs with dicut value noticeably larger than $m/2$ must have noticeable bias.

▶ **Theorem 4.** *There is a randomized streaming algorithm using $O(\log n)$ space that can, with probability $9/10$, distinguish between directed graphs with maximum dicut value more than $(1/2 + 8\epsilon)m$ from graphs with maximum dicut value at most $(1/4 + \epsilon)m$ (where m is the number of edges), for any $\epsilon \in (0, 1/16)$.*

1.3 Streaming complexity of Maximum Acylic Subgraph

In the final part of the paper, we turn to another fundamental problem, *Maximum Acyclic Subgraph* (MAS): Given a directed graph $G = (V, E)$, find an acyclic subgraph with maximum possible number of edges. Equivalently, we want an ordering of the vertices in V so that a maximum number of arcs in E go forward. Note that this makes MAS also a kind of 2CSP, albeit over a large domain $D = \{1, 2, \ldots, |V|\}$ with constraints of the form $x < y$.

The trivial algorithm which orders elements randomly, or the deterministic algorithm that takes the better solution among an arbitrary ordering and its reversal, achieves a factor 1/2 approximation. Unlike 2CSPs over fixed domains, where there are algorithms that beat the random assignment threshold [3, 14], for MAS there is no known polynomial time factor $(1/2 + \epsilon)$ approximation algorithm. However, such an algorithm is ruled out under Khot's Unique Games Conjecture [9]. The best known NP-hardness for MAS seems to be for approximation factors exceeding 65/66 [18].

Motivated by this state of affairs, we investigate whether one can show better hardness results against the restricted model of single-pass streaming algorithms. Our ultimate goal here would be to show that getting a $(1/2 + \epsilon)$-approximation requires polynomial space (we conjecture this to be the case). In this work, we prove the following weaker result. The proof proceeds via a reduction from the Boolean Hidden Matching problem, inspired by an analogous reduction for Max Cut from [16].

▶ **Theorem 5.** *Any randomized algorithm that, given a single pass over a stream of edges of an n-vertex directed graph G in arbitrary order, outputs a $(7/8+\epsilon)-$approximation to the MAS value of G with probability at least $3/4$, must use $\Omega_\epsilon(\sqrt{n})$ space.*

We note that the above hardness factor is much better than the currently best known NP-hardness. This raises a general theme of showing space lower bounds for approximation in the streaming model for problems that currently lack intractability results in the form of NP-hardness (or perhaps even Unique Games-hardness). In this broader context, one should of course take hardness results in the streaming model with a grain of salt – the streaming lower bound for Max Cut shows that streaming algorithms might be much weaker than polynomial time algorithms. Still, we view this direction as an interesting blend between approximation algorithms in general and constraint satisfaction in particular and streaming complexity, one that could nevertheless shed some new light on the core difficulty of problems such as MAS.

1.4 Open problems

We close this front matter by highlighting two natural open problems raised by our work.

1. What is the best approximation ratio achievable by a single-pass streaming algorithm with logarithmic space for Max 2CSP (or even the restricted Max DICUT)? The answer lies in the range $[2/5, 1/2]$. We suspect either $2/5$ or $1/2$ might be the right answer.
2. What is the best approximation ratio achievable by a single-pass streaming algorithm with logarithmic space for Maximum Acyclic Subgraph? The answer lies in the range $[1/2, 7/8]$. Here we conjecture that $1/2$ is the right answer.

2 Preliminaries

Max 2AND. We formally define the Max 2AND problem. An instance of the problem consists of a set of boolean variables $x_1, x_2, \ldots, x_n$, along with a set of clauses on these variables. Each clause consists of a conjunction of two literals, i.e., each clause is of one of the following forms, for some $i \neq j$: $x_i \wedge x_j$, $x_i \wedge \overline{x_j}$, or $\overline{x_i} \wedge \overline{x_j}$. The value of the instance is the maximum possible number of clauses that are satisfied for some setting of $x_1, x_2, \ldots, x_n$.

For each variable, it will be convenient to consider the number of constraints in which that variable appears as a literal in either positive or negative form. Thus, for any i, we define pos_i to be the number of constraints in which x_i appears non-negated, while we define neg_i to be the number of constraints in which x_i appears negated, i.e., as $\overline{x_i}$.

Special Case: Maximum Dicuts. One special case of the Max 2AND problem is the Max DICUT problem. We describe the Max DICUT in the terminology of graph theory below.

$G = (V, E)$ denotes a directed graph with vertex set V and edge set E, where $|V| = n$ and $|E| = m$. For any vertex $v \in V$, d_v, in_v, and out_v denote the overall degree, in-degree and out-degree of vertex v, respectively.

▶ **Definition 6.** A *dicut* is an ordered partition (A, B) of the vertex set of a directed graph into two disjoint subsets. The *dicut value* or *size* of the dicut is defined as the number of directed edges going from a vertex in A to a vertex in B.

▶ **Definition 7.** A *maximal dicut* (*Max DICUT*) of a directed graph $G = (V, E)$ is a dicut with the maximum dicut value.

Let (S, T) be an ordered partition of V and u be a vertex in V. Then, $E(S \to T)$ denotes the set of edges going from set S to set T, $E(u \to T)$ denotes the set of edges going from vertex u to vertices in set T, $E(S \to u)$ denotes the set of edges going from vertices in set S to vertex u, and $E(S \to S)$ denotes the edges with both endpoints inside the set S.

▶ Remark. Note that the Max DICUT problem can be viewed as a special case of the MAX 2AND problem in which each clause has exactly one positive literal and one negative literal. Vertices of the underlying graph correspond to boolean variables, and each directed edge from vertex i to vertex j corresponds to a clause of the form $x_i \wedge \overline{x_j}$. It is easy to see that a maximal dicut (A, B) in the graph terminology corresponds to the assignment of variables defined by $x_i = 1$ if vertex i is in set A, while $x_i = 0$ if vertex i is in set B.

▶ Remark. The value of any Max 2AND instance with m clauses is at least $\frac{m}{4}$, since a uniformly random assignment of boolean variables satisfies $\frac{m}{4}$ clauses on expectation.

▶ **Definition 8.** A randomized algorithm is said to give a $\alpha-$*approximation* to *Max 2AND* with *failure probability* δ (or *success probability* $1 - \delta$) if for any instance Ψ, it outputs a value in the interval $[\alpha d, d]$ with probability at least $1 - \delta$, where $\delta \in \left[0, \frac{1}{2}\right)$, $\alpha \in (0, 1)$, and d is the *Max 2AND* value of Ψ.

3 Single-Pass Streaming Complexity

Given a single pass over a stream of m constraints (in arbitrary order) of a Max 2AND instance Ψ over n variables, the problem is to estimate the Max 2AND value of Ψ using $O(\log n)$ space.

3.1 $(2/5 - \gamma)$-Approximation of Max 2AND

In analysing the Max 2AND value of an instance, it will be useful to consider a notion we call *bias*. Intuitively, for each vertex, we wish to compare the number of constraints in which x_i appears in positive form versus the number of constraints in which x_i appears in negated form. The following definition of bias captures this intuition.

▶ **Definition 9.** The *bias* of a Max 2AND instance Ψ on n variables, denoted bias_Ψ, is defined as

$$\text{bias}_\Psi = \sum_{i=1}^{n} |\text{pos}_i - \text{neg}_i|.$$

▶ Remark. $0 \leq \text{bias}_\Psi \leq 2m$.

Next, we prove a couple of theorems showing the relation between the bias of an instance and the Max 2AND value.

Intuitively, observe that if the bias of an instance is close to $2m$, then most variables x_i satisfy the property that most constraints involving x_i have the same literal on x_i (i.e., x_i appears in positive or negated form). Thus, it is reasonable to expect that in order to maximize the number of satisfied constraints, x_i should be set to the value that guarantees the truth of most of these literals. The following theorem essentially states that this is the case, and a bias close to $2m$ implies a Max 2AND value that is close to optimal, i.e., close to m.

▶ **Theorem 10.** *If the bias of an instance Ψ with n variables and m constraints is at least $(1-\epsilon)2m$, where $\epsilon \in [0,1]$, then the* Max 2AND *value of Ψ is at least $(1-\epsilon)m$.*

Proof. Assume that $\text{bias}_\Psi \geq (1-\epsilon)2m$. Now, consider the following greedy assignment $x_1 = x'_1, x_2 = x'_2, \ldots, x_n = x'_n$: For each i, we let $x'_i = 1$ if $\text{pos}_i \geq \text{neg}_i$, and $x'_i = 0$ otherwise. We claim that the number of constraints satisfied by this assignment is at least $(1-\epsilon)m$, which would imply that the Max 2AND value of Ψ is also at least $(1-\epsilon)m$.

Note that the number of unsatisfied constraints is at most

$$\sum_{i=1}^{n} \min\{\text{pos}_i, \text{neg}_i\}.$$

Thus, using the fact that

$$\begin{aligned}\text{bias}_\Psi &= \sum_{i=1}^{n} |\text{pos}_i - \text{neg}_i| \\ &= \sum_{i=1}^{n} (\text{pos}_i + \text{neg}_i) - 2\sum_{i=1}^{n} \min\{\text{pos}_i, \text{neg}_i\} \\ &= 2m - 2\sum_{i=1}^{n} \min\{\text{pos}_i, \text{neg}_i\}, \qquad (1)\end{aligned}$$

we have that the number of satisfied constraints of the assignment is at least

$$m - \sum_{i=1}^{n} \min\{\text{pos}_i, \text{neg}_i\} \geq m - \frac{2m - \text{bias}_\Psi}{2} = \frac{\text{bias}_\Psi}{2} \geq (1-\epsilon)m \ ,$$

as desired. ◀

The following theorem essentially shows a converse statement, namely, that in order to have a near-optimal Max 2AND value, i.e., close to m, the bias needs to be close to $2m$.

▶ **Theorem 11.** *If the bias of a Max 2AND instance Ψ with n variables and m constraints is at most $(1-\epsilon)2m$, where $\epsilon \in [0,1]$, then its Max 2AND value is at most $\left(1 - \frac{\epsilon}{2}\right)m$.*

Proof. Consider an assignment $x_1 = x'_1, x_2 = x'_2, \ldots, x_n = x'_n$ that satisfies the maximum number of constraints of Ψ. Note that for any i, at least $\min\{\text{pos}_i, \text{neg}_i\}$ constraints involving x_i are not satisfied. Therefore, the total number of constraints of Ψ that are not satisfied is at least

$$\frac{\sum_{i=1}^{n} \min\{\text{pos}_i, \text{neg}_i\}}{2}.$$

By (1), it follows that the Max 2AND value of Ψ is at most

$$\begin{aligned} m - \frac{\sum_{i=1}^{n} \min\{\text{pos}_i, \text{neg}_i\}}{2} &= m - \frac{2m - \text{bias}_\Psi}{4} = \frac{m}{2} + \frac{\text{bias}_\Psi}{4} \\ &\leq \frac{m}{2} + \frac{(1-\epsilon)2m}{4} = \left(1 - \frac{\epsilon}{2}\right)m, \end{aligned}$$

as desired. ◀

The above theorems show us that the bias of an instance and its Max 2AND value are closely related. Thus, if we can compute the bias of an instance efficiently in the single-pass streaming setting, then we obtain a method to estimate its Max 2AND value.

Algorithm 1 A $(2/5-\gamma)$-approximation algorithm of Max 2AND in the single-pass streaming setting.

```
Input: A single pass over the $m$ constraints of an instance $\Psi$ over $n$ variables $x_1, x_2, \ldots, x_n$, along with a parameter $\gamma < 2/5$ for desired closeness of approximation ratio.
Choose $\delta = 5\gamma/(4-5\gamma)$.
Compute the $L_1$ norm of the bias vector using the technique given in [15] (for an approximation within $1 \pm \delta$) to obtain $\widetilde{\text{bias}}_\Psi$.
if $\widetilde{\text{bias}}_\Psi \geq m/2$ then
    return $\widetilde{\text{bias}}_\Psi/2(1+\delta)$
else
    return $m/4$
end if
```

Let us define the *bias vector* of a Max 2AND instance Ψ with n variables and m constraints to be a vector with n components such that the i^{th} component is equal to $\text{pos}_i - \text{neg}_i$ for all $v \in V(G)$. Then, bias_Ψ is the L_1 norm of the bias vector. Each constraint $l_1 \wedge l_2$ that arrives in the stream changes the i^{th} and j^{th} components of the bias vector, where literal l_1 involves variable x_i and l_2 involves variable x_j. In particular, the arrival of the constraint increases the i^{th} component by 1 if l_1 is x_i while it decreases the component by 1 if l_1 is $\overline{x_i}$. Similarly, the j^{th} component increases by 1 if l_2 is x_j, while it decreases by 1 if l_2 is $\overline{x_j}$.

The following theorem given by Indyk in [15] shows that it is possible to compute the L_1 norm of a vector efficiently under bounded dynamic updates of its coordinates in the single-pass streaming setting.

▶ **Theorem 12** (from [15]). *Let S be a stream of data, where each chunk of data is of the form (i, a), $i \in [n]$ and $a \in \{-M \ldots M\}$, where M is a constant. The L_1 norm of the data defined by $L_1(S) = \|V(S)\|_1$, where $V(S)_i = \sum_{(i,a)\in S} a$ can be estimated by an algorithm that, given an arbitrary input stream S, outputs a quantity in the interval $[(1-\epsilon)L_1(S), (1+\epsilon)L_1(S)]$ with probability at least $9/10$, such that the algorithm uses only $O(\log n/\epsilon^2)$ words of storage.*

Theorem 12 can be adapted to our setting by converting each constraint of the form $x_i \wedge x_j$ to a data chunk $\{(i,1),(j,1)\}$, each constraint of the form $x_i \wedge \overline{x_j}$ to a data chunk $\{(i,1),(j,-1)\}$, and each constraint of the form $\overline{x_i} \wedge \overline{x_j}$ to a data chunk $\{(i,-1),(j,-1)\}$. This shows that we can compute the bias of a directed graph up to any constant precision with high probability.

We are now ready to show our main algorithmic result, namely, that one can obtain a 2/5-approximation to Max 2AND in the streaming model.

▶ **Theorem 13.** *Algorithm 1 is a $(2/5-\gamma)$-approximation algorithm of* Max 2AND *with success probability* $9/10$ *in the single-pass streaming setting.*

Proof. Note that if $\text{bias}_\Psi = (1/4+\epsilon)2m$, $\epsilon \in [\delta/2, 3/4]$, then by Theorem 10, the Max 2AND value Val of Ψ is at least $(1/4+\epsilon)m$ and by Theorem 11, Val is at most $(5/8+\epsilon/2)m$. Moreover, by Theorem 12, with probability at least 9/10, the L_1 norm estimation subroutine of Algorithm 1 outputs an estimate $\widetilde{\text{bias}}_\Psi \in ((1-\delta)\text{bias}_\Psi, (1+\delta)\text{bias}_\Psi)$. Since

$$(1-\delta)\text{bias}_\Psi \geq (1-\delta)\left(\frac{1}{4}+\epsilon\right)2m \geq (1-\delta)\left(\frac{1}{4}+\frac{\delta}{2}\right)2m \geq \frac{m}{2}\ ,$$

Algorithm 1 returns $\widetilde{\text{bias}_\Psi}/2(1+\delta)$ in this case. Furthermore,

$$\frac{\widetilde{\text{bias}_\Psi}/2}{\text{Val}} \geq \frac{(1-\delta)\text{bias}_\Psi/2}{(5/8+\epsilon/2)m} = \frac{(1-\delta)(1/4+\epsilon)m}{(5/8+\epsilon/2)m} \geq \frac{2}{5}(1-\delta)$$

and

$$\frac{\widetilde{\text{bias}_\Psi}/2}{\text{Val}} \leq \frac{(1+\delta)\text{bias}_\Psi/2}{(1/4+\epsilon)m} \leq \frac{(1+\delta)(1/4+\epsilon)m}{(1/4+\epsilon)m} \leq 1+\delta \ .$$

Thus, we obtain an approximation ratio of

$$\frac{2}{5} \cdot \frac{1-\delta}{1+\delta} \geq \frac{2}{5} - \gamma.$$

by the choice of δ in the algorithm

Next, consider the case in which $\text{bias}_\Psi < (1/4+\delta/2)2m$. Then, note that the algorithm always outputs a value that is at least $m/4(1+\delta)$. Moreover, by Theorem 11, we have that $\text{Val} \leq (5/8+\delta/4)m$. Therefore, the approximation ratio in this case is

$$\frac{m/4(1+\delta)}{(5/8+\delta/4)m} = \frac{2}{5} \cdot \frac{1+\delta}{1+\frac{2\delta}{5}} \geq \frac{2}{5} - \gamma \ . \qquad ◀$$

3.2 Hardness of $(1/2+\epsilon)$-approximation and a complementary streaming algorithm for Max DICUT

Next, we consider the Max DICUT problem. We start by examining the regime in which the Max DICUT value of an instance is close to the lower bound of $m/4$, i.e., as suboptimal as possible. For the Max DICUT problem, we define the following notion of bias for a directed graph G.

▶ **Definition 14.** We define the *bias* of a directed graph $G = (V, E)$, denoted bias_G, as $\text{bias}_G = \sum_{v \in V} |\text{out}_v - \text{in}_v|$.

▶ Remark. Note that bias_G, as defined in Definition 14, gives the identical value as bias_Ψ for the corresponding MAX 2AND instance Ψ (see Remark 2 for the correspondence). We use the notion bias_G along with the graph formulation of Max DICUT for convenience in this section.

The following theorem shows that if the DICUT value of a Max DICUT instance is close to $m/4$, then the bias of the corresponding graph must be small.

▶ **Theorem 15.** *If the Max DICUT value of a directed graph $G = (V, E)$ is at most $\left(\frac{1}{4}+\epsilon\right)m$, where $\epsilon \in \left[0, \frac{1}{16}\right]$, then its bias is at most $32\epsilon m$.*

Proof. Let $G = (V, E)$ be a directed graph with Max DICUT value at most $\left(\frac{1}{4}+\epsilon\right)m$, where $\epsilon \in \left[0, \frac{3}{4}\right]$. Let (A, B) be a maximal dicut of G.

$$|E(A \to B)| \leq \left(\frac{1}{4}+\epsilon\right)m. \tag{2}$$

$$|E(B \to A)| \leq \left(\frac{1}{4}+\epsilon\right)m. \tag{3}$$

For every vertex $u \in A$, we have

$$|E(u \to B)| \geq |E(A \to u)|. \tag{4}$$

If there is a $u \in A$ that does not satisfy (4), then it can be moved to set B to give a dicut of larger size, which contradicts the fact that (A, B) is a maximal dicut of G. Similarly, for every vertex $v \in B$, we have

$$|E(A \to v)| \geq |E(v \to B)|. \tag{5}$$

The total number of edges in the graph is

$$|E(A \to A)| + |E(B \to B)| + |E(A \to B)| + |E(B \to A)| = m. \tag{6}$$

From (4) and (5), we have

$$\max(|E(A \to A)|, |E(B \to B)|) \leq |E(A \to B)| \leq \left(\frac{1}{4} + \epsilon\right) m. \tag{7}$$

From (2), (3), (6) and (7), we get

$$\min(|E(A \to A)|, |E(B \to A)|, |E(B \to B)|) \geq \left(\frac{1}{4} - 3\epsilon\right) m. \tag{8}$$

We will now obtain an upper bound on $\sum_{v \in B} |\text{out}_v - \text{in}_v|$.

$$\begin{aligned}\sum_{v \in B} |\text{out}_v - \text{in}_v| &= \sum_{v \in B} \big||E(v \to B)| + |E(v \to A)| - |E(B \to v)| - |E(A \to v)|\big| \\ &\leq \sum_{v \in B} \left(\big||E(v \to B)| - |E(A \to v)|\big| + \big||E(v \to A)| - |E(B \to v)|\big|\right). \end{aligned} \tag{9}$$

Using (2),(5) and (8), we get

$$\begin{aligned}\sum_{v \in B} \big||E(v \to B)| - |E(A \to v)|\big| &= \sum_{v \in B} |E(A \to v)| - \sum_{v \in B} |E(v \to B)| \\ &\leq \left(\frac{1}{4} + \epsilon\right) m - \left(\frac{1}{4} - 3\epsilon\right) m \\ &= 4\epsilon m. \end{aligned} \tag{10}$$

We call a vertex $v \in B$ “good" if

$$|E(B \to v)| > |E(v \to A)|.$$

and “bad" if it is not “good". Now, consider the ordered partition (B, A). If we move a “good" vertex from B to A, the dicut value of the resulting partition is larger than the dicut value of (B, A). We know that the size of any dicut in G is at most $\left(\frac{1}{4} + \epsilon\right) m$. From (8), we can infer that the increase in the dicut value by moving a “good" vertex cannot exceed $4\epsilon m$. Let B_{g} denote the set of all “good" vertices in B. We have

$$\sum_{v \in B_{\text{g}}} (|E(B \to v)| - |E(v \to A)|) \leq 4\epsilon m. \tag{11}$$

Let B_{b} denote the set of all “bad" vertices in B. From (8) and (11), we get

$$\begin{aligned}\sum_{v \in B_{\text{b}}} |E(B \to v)| &\geq \left(\frac{1}{4} - 3\epsilon\right) m - \sum_{v \in B_{\text{g}}} |E(B \to v)| \\ &\geq \left(\frac{1}{4} - 3\epsilon\right) m - 4\epsilon m - \sum_{v \in B_{\text{g}}} |E(v \to A)| \\ &= \left(\frac{1}{4} - 7\epsilon\right) m - \sum_{v \in B_{\text{g}}} |E(v \to A)|. \end{aligned} \tag{12}$$

Using (12), we get

$$\begin{aligned}\sum_{v\in B_b}(|E(v\to A)|-|E(B\to v)|) &\le \sum_{v\in B}|E(v\to A)|-\left(\frac{1}{4}-7\epsilon\right)m\\ &= |E(B\to A)|-\left(\frac{1}{4}-7\epsilon\right)m\\ &\le 8\epsilon m. \qquad (13)\end{aligned}$$

From (9), (10), (11) and (13), we have

$$\sum_{v\in B}|\mathrm{out}_v-\mathrm{in}_v|\le 16\epsilon m.$$

Using similar arguments, we can conclude that

$$\sum_{v\in A}|\mathrm{out}_v-\mathrm{in}_v|\le 16\epsilon m.$$

Hence, the bias of G is at most $32\epsilon m$. ◀

▶ **Corollary 16.** *If the* Max DICUT *value of a directed graph* $G=(V,E)$ *is* $\frac{m}{4}$*, then its bias is* 0*, i.e.,* $\mathrm{in}_v=\mathrm{out}_v\ \ \forall v\in V$*.*

▶ Remark. It is reasonable to expect a converse statement of Theorem 15 to hold, i.e., that a bias close to zero implies a Max DICUT value that is close to $m/4$. However, it turns out that such a statement is not true. For example, consider the following instance of Max DICUT: Let $G(V,E)$ be an undirected perfect matching on $2n$ vertices. Using G, construct a directed graph $G'(V,E')$ by adding directed edges $u\to v,\ \ v\to u$ to E' for each undirected edge (u,v) in E. The Max DICUT value of G' is n, which is much larger than $m/4=2n/4=n/2$. However, the bias of G' is 0.

We now state the non-existence of a better-than-1/2−approximation algorithm for Max DICUT when the constraints of an instance arrive one-by-one in random order in the single-pass streaming setting.

Kapralov et al. gave a lower bound for approximating MAXCUT in the single-pass streaming setting in [16]. By showing a simple reduction from MAXCUT to Max DICUT, we observe that the same lower bound applies to Max DICUT. The following theorem is taken from [16].

▶ **Theorem 17** (from [16]). *Let* $\epsilon>0$ *be a constant. Let* $G=(V,E)$*,* $|V|=n$*,* $|E|=m$ *be an undirected graph. Any randomized algorithm that, given a single pass over a stream of edges of* G *presented in random order, outputs a* $(1/2+\epsilon)-$*approximation to the value of the maximum cut in* G *with probability at least* $9/10$ *over its internal randomness must use* $\tilde{\Omega}(\sqrt{n})$ *space.*

The reduction from MAXCUT to Max DICUT is as follows. Given any undirected graph G, convert it into a directed graph G' by adding two directed edges $u\to v$ and $v\to u$ for every edge $(u,v)\in E(G)$. Observe that G has a cut of size k if and only if G' has a dicut of size k. Therefore, the MAXCUT value of G is equal to the Max DICUT value of G'. Note that this reduction can be done on the fly and does not require any additional storage. Thus, we have the following theorem.

▶ **Theorem 18.** *Let $\epsilon > 0$ be a constant. Let $G = (V, E)$, $|V| = n$, $|E| = m$ be a directed graph. Any randomized algorithm that, given a single pass over a stream of edges of G presented in random order, outputs a $(1/2 + \epsilon)-$approximation to the* Max DICUT *value of G with probability at least $9/10$ over its internal randomness must use $\tilde{\Omega}(\sqrt{n})$ space.*

▶ Remark. Since an instance of Max DICUT can be viewed as a special instance of Max 2AND in which all clauses have one positive literal and one negative literal, the aforementioned theorem also precludes the existence of a randomized streaming algorithm that approximates the *Max 2AND* value of an instance to a factor of $1/2 + \epsilon$ without using $\tilde{\Omega}(\sqrt{n})$ space.

To prove Theorem 17, [16] showed that no $o(\sqrt{n})$ algorithm can distinguish between distributions D^Y and D^N, where graphs drawn from D^Y have MAXCUT value m, while graphs drawn from D^N have MAXCUT value at most $(1/2 + \epsilon)m$ for any $\epsilon \in [0, 1/2]$. By applying the reduction from MAXCUT to Max DICUT (the number of edges is doubled in the reduced graph), we can conclude than no $o(\sqrt{n})$ algorithm can distinguish between directed graphs with Max DICUT value $m/2$ and graphs with Max DICUT value at most $(1/4 + \epsilon)m$ for any $\epsilon \in [0, 1/4]$.

▶ **Theorem 19.** *Given a single pass over the edges of a directed graph G in any order, by computing the bias of G we can distinguish between directed graphs with* Max DICUT *value more than $(1/2 + 8\epsilon)m$ and graphs with* Max DICUT *value at most $(1/4 + \epsilon)m$ for any $\epsilon \in (0, 1/16)$, with success probability $9/10$.*

Proof. If the Max DICUT value of a graph is at most $(\frac{1}{4} + \epsilon)m$, then using Theorem 15 we can conclude that its bias is at most $32\epsilon m$. If the Max DICUT value of a graph is more than $(1 - \delta)m$, then using the contrapositive result of Theorem 11, we can conclude that its bias is more than $(1 - 2\delta)2m$. By substituting $\delta = 1/2 - 8\epsilon$, we get that the bias of the graph is more than $32\epsilon m$. Thus, we can distinguish the two graphs by computing their bias values using the L_1 sampling method. ◀

Note that while [16] implied that no $o(\sqrt{n})$ algorithm can distinguish between directed graphs with Max DICUT value $m/2$ and graphs with Max DICUT value at most $(1/4+\epsilon)m$ for any $\epsilon \in [0, 1/4]$, Theorem 19 shows that it is possible to distinguish between directed graphs with Max DICUT value $(1/2 + 0.0001)m$ and graphs with Max DICUT value at most $(1/4 + 0.00001)m$ in the single-pass streaming setting using a $O(\log n)$ algorithm that computes the bias of a graph.

4 Maximum Acyclic Subgraph

▶ **Definition 20.** The Maximum Acyclic Subgraph (MAS) value of a directed graph $G = (V, E)$ is the size of the largest acyclic subgraph of G, where we define the size of a graph to be the number of edges in it.

Given a single pass over a stream of edges of a directed graph $G = (V, E)$, we are interested in computing an approximate estimate of the MAS value of G using logarithmic space.

▶ **Definition 21.** A randomized algorithm is said to give a $\alpha-$approximation to MAS for some $\alpha \in (0, 1)$ if for any input $G = (V, E)$, it outputs a value in the interval $[\alpha d, d]$ with probability at least $9/10$, where d is the MAS value of G.

In this section, we show the non-existence of a better-than-7/8−approximation algorithm to MAS in the low-space single-pass streaming setting. We show this via a reduction from

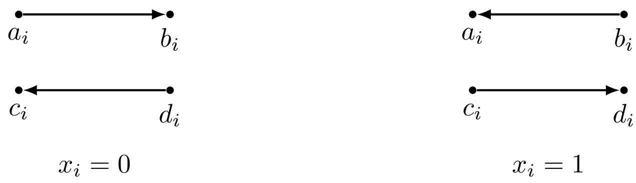


Figure 1 Edge set E_1.

Boolean Hidden Matching (BHM), a two party one-way communication problem. A strong communication lower bound for BHM was given by Gavinsky et al. in [6]. In [16], Kapralov et al. showed the hardness of approximating MAXCUT using the BHM problem and its extension given by Verbin and Yu in [22]. Inspired by this, here we adapt this approach to show hardness of approximating MAS.

▶ **Definition 22.** The Boolean Hidden Matching (*BHM*) problem is a communication complexity problem in which Alice gets a Boolean vector $x \in \{0,1\}^n$ and Bob gets an undirected perfect matching M on n vertices and a Boolean vector w of length $n/2$, where we identify the perfect matching M with its Boolean edge incidence matrix of dimension $\frac{n}{2} \times n$. It is a promise problem in which Bob outputs **YES** when $Mx \oplus w = 0^{n/2}$ and outputs **NO** when $Mx \oplus w = 1^{n/2}$.

The following theorem was proved by Gavinsky et al. in [6].

▶ **Theorem 23.** *Any randomized one-way communication protocol for solving* BHM*, where Alice sends messages to Bob, that succeeds with probability at least* $9/10$ *has complexity* $\Omega(\sqrt{n})$*.*

We now use the above theorem to prove our streaming lower bound for approximating MAS.

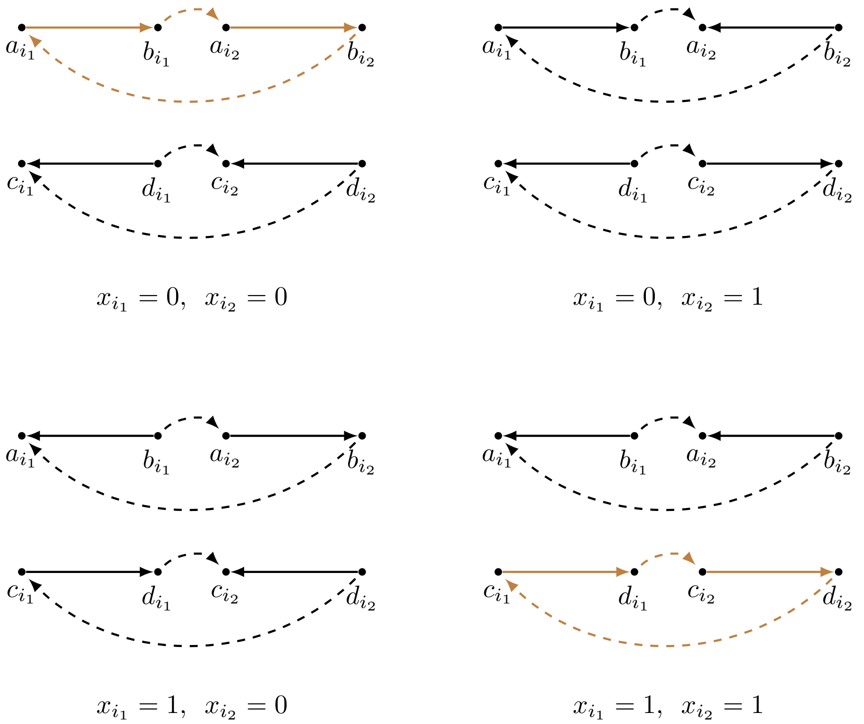
▶ **Theorem 24.** *Let $\epsilon > 0$ be a constant. Let $G = (V, E)$, $|V| = n$, $|E| = m$ be a directed graph. Any randomized algorithm that, given a single pass over a stream of edges of G, outputs a $(7/8 + \epsilon)-$approximation to the MAS value of G with probability at least $9/10$ over its internal randomness must use $\Omega(\sqrt{n})$ space.*

Proof. Let **ALG** be a randomized algorithm that uses space c and gives a better-than-$7/8-$approximation to MAS in the single-pass streaming setting. We will show that **ALG** can be used to obtain a randomized one-way communication protocol for BHM with complexity c.

Let $x \in \{0,1\}^n$ be the vector that Alice receives. Alice creates edge set E_1, her part of the graph that will be given as input to **ALG**, as shown in Fig. 1. For each $i \in [n]$, she creates four vertices a_i, b_i, c_i and d_i. If $x_i = 0$, she adds edges $a_i \to b_i$ and $d_i \to c_i$ to E_1. Else, she adds edges $b_i \to a_i$ and $c_i \to d_i$ to E_1. She then treats E_1 as the first half of the stream of edges, runs **ALG** on E_1 and sends the state of **ALG** to Bob.

Bob constructs edge set E_2, his part of the graph, as shown in Fig. 2 and Fig. 3. Let M be the perfect matching on n vertices and w be the boolean vector of length $n/2$ that Bob receives. Let $M_i = (i_1, i_2)$ denote the $i-$th edge in the matching M (fix any ordering) and w_i denote the $i-$th coordinate of w, where $i \in [n/2]$. If $w_i = 0$, he adds edges $b_{i_1} \to a_{i_2}$, $b_{i_2} \to a_{i_1}$, $d_{i_1} \to c_{i_2}$ and $d_{i_2} \to c_{i_1}$ to E_2. Else, he adds edges $b_{i_1} \to b_{i_2}$, $a_{i_2} \to a_{i_1}$, $d_{i_1} \to d_{i_2}$ and $c_{i_2} \to c_{i_1}$ to E_2.

He treats E_2 as the second half of the stream and completes the execution of **ALG** on the stream starting from the state that Alice sent. The total number of edges in the graph is



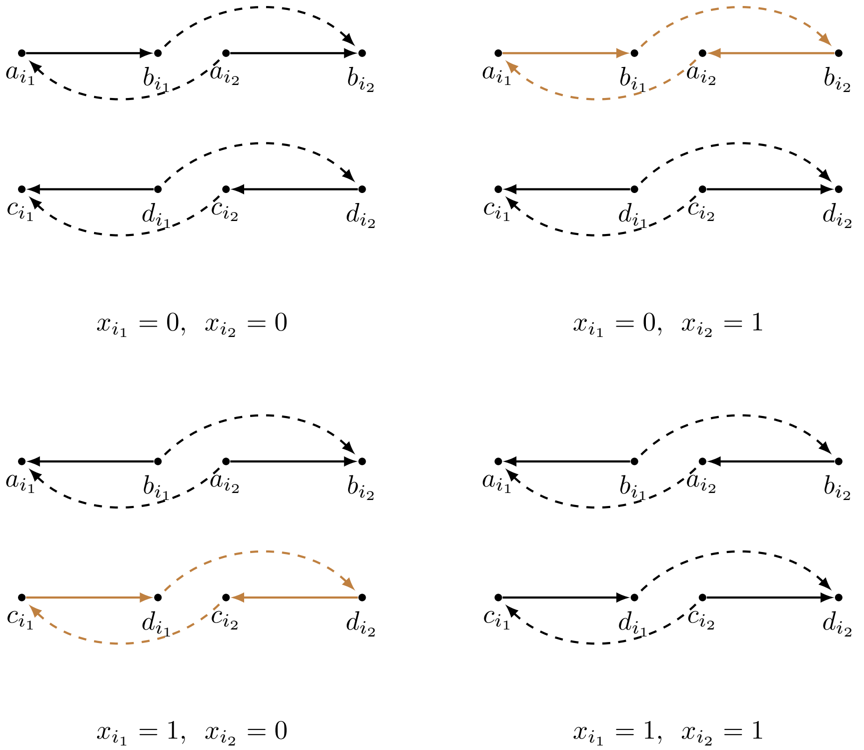
$x_{i_1} = 0, \ \ x_{i_2} = 0$

$x_{i_1} = 0, \ \ x_{i_2} = 1$

$x_{i_1} = 1, \ \ x_{i_2} = 0$

$x_{i_1} = 1, \ \ x_{i_2} = 1$

Figure 2 Edge set E_2 when $w_i = 0$ (Cycles are marked in brown).



$x_{i_1} = 0, \ \ x_{i_2} = 0$

$x_{i_1} = 0, \ \ x_{i_2} = 1$

$x_{i_1} = 1, \ \ x_{i_2} = 0$

$x_{i_1} = 1, \ \ x_{i_2} = 1$

Figure 3 Edge set E_2 when $w_i = 1$ (Cycles are marked in brown).

$2n + 4(n/2) = 4n$ (Alice adds two edges for each coordinate of x and Bob adds four edges for each edge in M). If the MAS value output by **ALG** is greater than $7n/2$, then Bob outputs **NO**, else he outputs **YES**.

The correctness of the reduction can be shown in the following way.

$$Mx \oplus w = \begin{bmatrix} x_{1_1} \oplus x_{1_2} \oplus w_1 \\ \vdots \\ x_{i_1} \oplus x_{i_2} \oplus w_i \\ \vdots \\ x_{(n/2)_1} \oplus x_{(n/2)_2} \oplus w_{(n/2)} \end{bmatrix}.$$

As depicted in Fig. 2 and Fig. 3, we can observe that when $x_{i_1} \oplus x_{i_2} \oplus w_i = 0$, there is exactly one cycle and when $x_{i_1} \oplus x_{i_2} \oplus w_i = 1$, there are no cycles. Therefore, if $Mx \oplus w = 1^{n/2}$, then the graph is acyclic and the MAS value is $4n$. If $Mx \oplus w = 0^{n/2}$, then the graph contains exactly $n/2$ disjoint cycles (corresponding to each $i \in [n/2]$) and hence the MAS value is $7n/2$ (subtract one edge from each cycle to get the maximum acyclic subgraph). Since **ALG** gives a better-than-7/8−approximation to MAS, it outputs a MAS value greater than $7n/2$ if and only if $Mx \oplus w = 1^{n/2}$. Since the state sent by Alice to Bob (after executing the first half of the stream) contains at most c bits, the above protocol has randomized one-way communication complexity c with success probability at least 9/10. By using Theorem 23, we infer that $c = \Omega(\sqrt{n})$. ◀

References

1 Paola Alimonti. Non-oblivious local search for MAX 2-CSP with application to MAX DICUT. In *23rd International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 2–14, 1997.

2 András A. Benczúr and David R. Karger. Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. *Proceedings of the 28th annual ACM symposium on Theory of computing*, pages 47–55, 1996.

3 Lars Engebretsen and Venkatesan Guruswami. Is constraint satisfaction over two variables always easy? *Random Structures and Algorithms*, 25(2):150–178, 2004.

4 Uriel Feige and Michel X. Goemans. Aproximating the value of two prover proof systems, with applications to MAX 2SAT and MAX DICUT. In *Third Israel Symposium on Theory of Computing and Systems (ISTCS)*, pages 182–189, 1995. doi:10.1109/ISTCS.1995.377033.

5 Uriel Feige and Shlomo Jozeph. Oblivious algorithms for the Maximum Directed Cut problem. *Algorithmica*, 71(2):409–428, 2015. doi:10.1007/s00453-013-9806-z.

6 Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald de Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. In *Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing*, STOC'07, pages 516–525, New York, NY, USA, 2007. ACM. doi:10.1145/1250790.1250866.

7 Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995. doi:10.1145/227683.227684.

8 Venkatesan Guruswami and Euiwoong Lee. Towards a characterization of approximation resistance for symmetric CSPs. In *Proceedings of Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques (APPROX/RANDOM)*, pages 305–322, 2015. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.305.

9 Venkatesan Guruswami, Rajsekar Manokaran, and Prasad Raghavendra. Beating the random ordering is hard: Inapproximability of maximum acyclic subgraph. In *Proceedings of the 49th IEEE Symposium on Foundations of Computer Science*, pages 573–582, 2008.

10 Venkatesan Guruswami and Yuan Zhou. Tight bounds on the approximability of almost-satisfiable horn SAT and exact hitting set. *Theory of Computing*, 8(11):239–267, 2012. doi:10.4086/toc.2012.v008a011.

11 Eran Halperin and Uri Zwick. Combinatorial approximation algorithms for the maximum directed cut problem. In *Proceedings of the 12th Annual Symposium on Discrete Algorithms*, pages 1–7, 2001. URL: http://dl.acm.org/citation.cfm?id=365411.365412.

12 Eran Halperin and Uri Zwick. Combinatorial approximation algorithms for the maximum directed cut problem. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'01, pages 1–7, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=365411.365412.

13 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

14 Johan Håstad. Every 2-CSP allows nontrivial approximation. *Computational Complexity*, 17(4):549–566, 2008. doi:10.1007/s00037-008-0256-y.

15 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, May 2006. doi:10.1145/1147954.1147955.

16 Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Streaming Lower Bounds for Approximating MAX-CUT. In *Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'15, pages 1263–1282, Philadelphia, PA, USA, 2015. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=2722129.2722213.

17 Michael Kapralov, Sanjeev Khanna, Madhu Sudan, and Ameya Velingker. $(1+\Omega(1))$-approximation to MAX-CUT Requires Linear Space. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'17, pages 1703–1722, Philadelphia, PA, USA, 2017. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=3039686.3039798.

18 Alantha Newman. Approximating the maximum acyclic subgraph. Master's thesis, MIT, June 2000.

19 D. A. Spielman and N. Srivastava. Graph sparsification by effective resistances. *STOC*, pages 563–568, 2008.

20 Johan Thapper and Stanislav Zivny. The complexity of finite-valued CSPs. *J. ACM*, 63(4):37:1–37:33, 2016. doi:10.1145/2974019.

21 Luca Trevisan. Parallel approximation algorithms by positive linear programming. *Algorithmica*, 21(1):72–88, 1998. doi:10.1007/PL00009209.

22 Elad Verbin and Wei Yu. The streaming complexity of cycle counting, sorting by reversals, and other problems. In *Proceedings of the Twenty-second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'11, pages 11–25, Philadelphia, PA, USA, 2011. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=2133036.2133038.

A Approximating Max DICUT Using LP Rounding

In this section, we give a deterministic $1/2-$approximation algorithm to solve the Max DICUT problem in polynomial time in the usual setting (not streaming).

We first look at the binary integer programming (BIP) formulation of the Max DICUT problem. Given a directed graph $G = (V, E)$, the objective is to obtain an ordered partition

(A, B) of V such that the number of edges not in $E(A \to B)$ is minimum.

$$\begin{aligned} \text{minimize} \quad & \sum_{e(i,j)\in E} z_e. \\ \text{subject to} \quad & z_e \geq x_i, \qquad \forall e(i,j) \in E. \\ & z_e \geq 1 - x_j, \, \forall e(i,j) \in E. \\ & z_e, x_i \in \{0,1\}, \quad \forall i \in V, \forall e \in E. \end{aligned}$$

In the above BIP, $e(i, j)$ denotes a directed edge e from vertex i to vertex j. x_i, $\forall i \in V$ are indicator variables. If $x_i = 0$, then $i \in A$, else $i \in B$. z_e is constrained to be at least as large as $\max(x_i, 1 - x_j)$. Since the objective is to minimize $\sum_{e(i,j)\in E} z_e$, in any BIP solution, $z_e = \max(x_i, 1 - x_j)$. Thus, $z_e = 0$ if $e \in E(A \to B)$ and 1 otherwise. Hence, the optimal value of $\sum_{e(i,j)\in E} z_e$ gives the number of edges not in the Max DICUT of G.

It is hard to get an exact solution for the above BIP in the worst case. Therefore, we consider below its LP relaxation to approximately estimate the number of edges not in the Max DICUT.

$$\begin{aligned} \text{minimize} \quad & \sum_{e(i,j)\in E} z_e. \\ \text{subject to} \quad & z_e \geq x_i, \qquad \forall e(i,j) \in E. \\ & z_e \geq 1 - x_j, \, \forall e(i,j) \in E. \\ & z_e, x_i \in [0,1]\,, \quad \forall i \in V, \forall e \in E. \end{aligned}$$

Any optimum solution to the above LP assigns $z_e = \max(x_i, 1 - x_j)$. Thus $\{x_i\}$ are the independent variables and we denote any solution f to the above LP by $f = \{x_i\}$. The existence of a half-integral optimal solution to the LP and a combinatorial algorithm to obtain it was given by Halperin and Zwick in [12]. In this paper, we present an alternate algorithm to obtain a half-integral optimal solution to the LP, adapted from [10].

▶ **Theorem 25** (Half Integrality). *There is a polynomial-time algorithm that given a optimal solution $f = \{x_i\}$ to the above LP, converts f into another optimal solution $f^* = \{x_i^*\}$ such that each x_i^* is half-integral, i.e., $x_i^* \in \{0, 1, 1/2\}$, and $Val(f^*) \leq Val(f)$.*

Proof. Algorithm 2 takes as input the above LP formulation and one of the solutions $f = \{x_i\}$, and outputs the desired f^*. At a high level, the algorithm iteratively moves the LP variables that are not half integral to half integral values. We need to prove that the algorithm terminates in polynomial number of iterations and in each iteration, it creates a valid LP solution whose objective value is at most the previous objective value.

Algorithm 2 always maintains a valid solution f to the LP (i.e., all x_i's are in the range $[0, 1]$). We first prove that it terminates within a polynomial number of iterations. Consider the set $W_f = \{0 < x < 1/2 : \exists i \in V \mid x = x_i \text{ or } x = 1 - x_i\}$. In each loop, the algorithm picks a p from W_f. At the end of the loop, p is removed from W_f and no new element is added. Thus, after a linear number of steps $W_f = \emptyset$ and the loop terminates.

We now prove that the objective value of the LP does not increase after each iteration. Define $f^{(t)} = \{x_i^{(t)} = t\}_{i\in S} \cup \{x_i^{(t)} = 1 - t\}_{i\in S'} \cup \{x_i^{(t)} = x_i\}_{i\in V\setminus(S\cup S')}$ for $t \in [a, b]$. Observe that $p \in [a, b]$. If we can show that $\text{Val}(f^{(t)})$ is a linear function for $t \in [a, b]$, it proves that $\min(\text{Val}(f^{(a)}), \text{Val}(f^{(b)})) \leq \text{Val}(f^{(p)}) = \text{Val}(f)$. To prove linearity of $\text{Val}(f^{(t)})$, we only need to show that $g_{ij}(t) = \max(x_i^{(t)}, 1 - x_j^{(t)})$ is linear for $t \in [a, b]$, $\forall e(i, j) \in E$. We prove each case separately for $t \in [a, b]$.

Algorithm 2 Round any LP solution $f = \{x_i\}$ to a half-integral solution f^*, with $\text{Val}(f^*) \leq \text{Val}(f)$.

```
while $\exists i \in V : x_i \notin \{0, 1, 1/2\}$ do
    choose $k \in v$, such that $x_k \notin \{0, 1, 1/2\}$ (arbitrarily)
    if $x_k < 1/2$ then
        $p \leftarrow x_k$
    else
        $p \leftarrow 1 - x_k$
    end if
    $S \leftarrow \{i : x_i = p\}, S' \leftarrow \{i : x_i = 1 - p\}$
    $a \leftarrow \max\{x_i : x_i < p, 1 - x_i : x_i > 1 - p, 0\}$
    $b \leftarrow \min\{x_i : x_i > p, 1 - x_i : x_i < 1 - p, 1/2\}$
    $f^{(a)} \leftarrow \{x_i^{(a)} = a\}_{i \in S} \cup \{x_i^{(a)} = 1 - a\}_{i \in S'} \cup \{x_i^{(a)} = x_i\}_{i \in V \backslash (S \cup S')}$
    $f^{(b)} \leftarrow \{x_i^{(b)} = b\}_{i \in S} \cup \{x_i^{(b)} = 1 - b\}_{i \in S'} \cup \{x_i^{(b)} = x_i\}_{i \in V \backslash (S \cup S')}$
    if $\text{Val}(f^{(a)}) \leq \text{Val}(f^{(b)})$ then
        $f \leftarrow f^{(a)}$
    else
        $f \leftarrow f^{(b)}$
    end if
end while
return $f$ (as $f^*$)
```

- If $i, j \in V \backslash (S \cup S')$, $g_{ij}(t)$ is a constant function.
- If $i \in S, j \in S'$, $g_{ij}(t) = t$.
- If $i \in S', j \in S$, $g_{ij}(t) = 1 - t$.
- If $i, j \in S$ (or $i, j \in S'$), $g(t) = \max(t, 1 - t)$. Since $a, b \leq \frac{1}{2}$ and $t \in [a, b]$, $g_{ij}(t) = 1 - t$.
- If $i \in S, j \in V \backslash (S \cup S')$, $g(t) = \max(t, 1 - x_j)$. If we plot all the x_i's and $1 - x_i'$s on the $[0, 1]$ line, a is the maximum value less than p and b is the minimum value greater than p. $\forall i \in V \backslash (S \cup S'), x_i \notin (a, b)$ and $1 - x_i \notin (a, b)$.Thus, depending on x_j, $g_{ij}(t)$ is either a constant or a linear function.
- If $i \in S', j \in V \backslash (S \cup S')$ (or $i \in V \backslash (S \cup S'), j \in S \cup S'$), we can show that $g_{ij}(t)$ is linear by using the same argument as above. ◀

▶ **Lemma 26.** *If the optimum value of the LP is at most ϵm, then the* Max DICUT *value of the corresponding graph is at least $\left(1 - \frac{3}{2}\epsilon\right) m$.*

Proof. Using Algorithm 2, we obtain a half-integral solution to the LP relaxation in polynomial time. This solution partitions the vertex set into three subsets. Let $A = \{i : x_i = 0\}$, $B = \{i : x_i = 1\}$ and $U = \{i : x_i = 1/2\}$. The solution assigns $z_e = 0$ for $e \in E(A \to B)$, $z_e = 1/2$ for $e \in E(U \to B) \cup E(A \to U) \cup E(U \to U)$ and $z_e = 1$ otherwise. If we round off each variable in U to either 0 or 1 with probability 1/2, on expectation, at least half of $\{z_e\}$ that are assigned value 1/2 currently become 0. This implies that there exists a rounding r which makes at most half of $\{z_e\}$ with value 1/2 become 1 after that.

Since the LP optimum is at most ϵm, the number of $\{z_e\}$ that take value 1/2 are at most $2\epsilon m$. After rounding r, the LP solution looks similar to the BIP solution. The increase in the objective value is at most $\frac{1}{2} \times 2\epsilon m \times \frac{1}{2} = \frac{\epsilon}{2} m$. Thus, the Max DICUT value of the graph is at least $\left(1 - \frac{3}{2}\epsilon\right) m$. ◀

Algorithm 3 A deterministic $1/2-$approximation algorithm of Max DICUT.

```
Input: A directed graph $G = (V, E)$.
Solve the LP relaxation of the Max DICUT problem for $G$. Let $t$ be the corresponding
optimum value.
if $t \leq m/2$ then
    return $(m - 3t/2)$
else
    return $m/4$
end if
```

▶ **Theorem 27.** *Algorithm 3 is a deterministic polynomial time* $1/2-$*approximation algorithm of* Max DICUT*.*

Proof. The running time of Algorithm 3 follows from the fact that any LP can be solved in deterministic polynomial time. If t is the optimum value returned by the LP relaxation, then the BIP optimum value is at least t. This implies that the Max DICUT value of the corresponding graph is at most $m - t$. Lemma 26 implies that the Max DICUT value is at least $(m - 3t/2)$. When $t \leq m/2$, the algorithm returns $(m - 3t/2)$ as the Max DICUT value. In this case, the approximation ratio is $(m - 3t/2)/(m - t) \geq 1/2$. When $t > m/2$, the Max DICUT value is at most $m/2$. Since the algorithm outputs $m/4$ in this case, the approximation ratio is $1/2$. ◀

Symmetric Interdiction for Matching Problems

**Samuel Haney[1], Bruce Maggs[2], Biswaroop Maiti[3],
Debmalya Panigrahi[4], Rajmohan Rajaraman[5], and Ravi Sundaram[6]**

1 **Duke University, Durham, NC, USA**
shaney@cs.duke.edu
2 **Duke University, Durham, NC, USA; and
Akamai Technologies, Cambridge, MA, USA**
bmm@cs.duke.edu
3 **Northeastern University, Boston, MA, USA**
biswaroop@ccs.neu.edu
4 **Duke University, Durham, NC, USA**
debmalya@cs.duke.edu
5 **Northeastern University, Boston, MA, USA**
rraj@ccs.neu.edu
6 **Northeastern University, Boston, MA, USA**
koods@ccs.neu.edu

Abstract

Motivated by denial-of-service network attacks, we introduce the symmetric interdiction model, where both the interdictor and the optimizer are subject to the same constraints of the underlying optimization problem. We give a general framework that relates optimization to symmetric interdiction for a broad class of optimization problems. We then study the symmetric matching interdiction problem – with applications in traffic engineering – in more detail. This problem can be simply stated as follows: find a matching whose removal minimizes the size of the maximum matching in the remaining graph. We show that this problem is APX-hard, and obtain a 3/2-approximation algorithm that improves on the approximation guarantee provided by the general framework.

1998 ACM Subject Classification G.2.2 [Graph Theory] Graph Algorithms, Network Problems

Keywords and phrases Approximation algorithms, matching, interdiction

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.9

1 Introduction

A recent study of malicious network traffic observed at Microsoft data centers [17] made the surprising observation that a large volume of attack traffic originated from virtual machines hosted within the data centers themselves. The machines generating these attacks may have been compromised, or they may have been rented with stolen credit cards or on a free-trial basis. While the authors of the study used heuristics to identify traffic that was obviously malicious, in general it is very difficult to distinguish legitimate traffic from malicious traffic. In particular, an attacker in possession of a "botnet" of compromised machines can launch a denial-of-service attack against a service simply by using these machines to send a large number of legitimate-looking requests to the servers that implement the service.

The following question then arises: how does a network operator decide which connection requests to admit if she cannot distinguish between legitimate and malicious requests? One natural strategy is to minimize *regret*: the number of legitimate requests that are not served

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 9; pp. 9:1–9:19
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

but might have been otherwise. This motivates us to define the *symmetric interdiction* model in this paper, where the goal is to select a feasible set of edges whose removal minimizes the maximum feasible set in the remaining graph. We give a general framework for converting algorithms for a broad class of optimization problems to algorithms for the corresponding symmetric interdiction problems.

We instantiate our general model in the *symmetric matching interdiction* problem (abbreviated SMI in the rest of the paper), where the goal is to select a matching whose removal minimizes the maximum matching in the remaining graph. The SMI problem models our motivating scenario. Suppose clients located in a data center issue requests to servers in the same data center, where each client and each server has the capacity to participate in a single client-server interaction. Each client provides the operator of the data center with a list of servers it would like to contact, and the operator selects a matching of clients and servers. The operator would prefer to prioritize legitimate requests, but cannot distinguish between legitimate and malicious clients. By minimizing the size of the remaining maximum matching, an optimal solution to the SMI problem bounds the number of legitimate requests that are not satisfied but might otherwise have been. For the SMI problem, we show hardness results, and give a carefully designed algorithm that improves upon the result obtained from the general framework.

Main Results. Consider a generic optimization problem Π that is specified by an input graph $G = (V, E)$, by a set $\mathcal{F}$ of subgraphs of G which constitute feasible solutions to the problem, and a maximization (resp., minimization) objective function f on graphs. An example of Π is the maximum matching problem: $\mathcal{F}$ is the set of all matchings and the function f returns the number of edges in the matching. For the optimization problem Π, we define the symmetric interdiction problem $I(\Pi)$ as follows: the goal is to produce a subgraph $H = (V, F)$ of G such that H is in $\mathcal{F}$ and minimizes (resp., maximizes) the optimum value of f achievable on the remaining graph $(V, E \setminus F)$. Thus, the symmetric matching interdiction (SMI) problem is given a graph G and seeks a matching M of G so as to minimize the maximum matching in $G \setminus M$.

Our first result is a general framework for converting optimization algorithms to symmetric interdiction algorithms for a broad class of problems. This result, described informally below, is stated formally in Theorem 3 and proved in Section 2.

▶ **Theorem 1** (Informal). *An α-approximation to a packing problem Π implies a $(1 + \alpha)$-approximation to the corresponding symmetric interdiction problem $I(\Pi)$, modulo some technical conditions.*

Next, we focus on the SMI problem. Theorem 3 implies that any maximum matching algorithm is a 2-approximation algorithm for this problem. In fact, we show that any *maximal* matching also achieves an approximation factor of 2. However, this is the limit of the general framework in the sense that there are graphs where a maximum matching has an approximation factor of exactly 2 for the SMI problem. Our main algorithmic contribution is to obtain a more careful algorithm for the SMI problem that obtains an approximation factor of 1.5. We complement this result with a proof of APX-hardness of the problem by giving an approximation lower bound of $(1 + \epsilon)$ for small but fixed positive ϵ.

▶ **Theorem 2.** *There is a polynomial-time deterministic algorithm for the symmetric matching interdiction problem with an approximation factor of* 1.5*. Moreover, the symmetric matching interdiction problem is APX-hard.*

Extensions. We consider a randomized variant of the SMI problem in Section 5. Specifically, we show that if the interdictor is allowed to use randomness that is invisible to the optimizer, then the SMI problem becomes polynomial time solvable. Another natural extension of the SMI problem that captures practical parameters arising in networking is the capacitated case, where every edge has a capacity and the input and output ports have maximum capacities on the total amount of network flow that can be routed through them. On the other hand, if the edge capacities are unsplittable and both the interdictor's and optimizer's solutions are edge subsets, then the corresponding optimization problem is a special case of the previously studied *demand matching* problem [22]. Existing results for the optimization problem, applied to our interdiction framework, gives an approximation algorithm for the interdiction problem using Theorem 1. Improving this result using a more specific algorithm, such as the one that we give for the SMI problem, is left as an open problem in this work.

Finally, our symmetric interdiction framework can be applied to other diverse combinatorial optimization problems. See Section 6 for a brief discussions on other symmetric interdiction problems.

Related Work. Interdiction variants of classical graph optimization problems have attracted considerable research interest in recent years. Typically, these problems are modeled as a two-step game between an *interdictor* and an *optimizer*. In the first step, the interdictor removes a limited number of edges from the graph, with the goal of worsening the objective of the optimizer who solves the graph optimization problem on the remaining graph in the second step. For instance, in the matching interdiction problem, the goal is to remove at most k edges (for a given k) such that the size of the maximum matching in the remaining graph is minimized [26]. One can similarly define interdiction variants for maximum flow [7, 9, 10, 25, 27, 6, 1, 5, 16], minimum spanning tree [28, 8], and many other classic graph optimization problems [4, 23, 12, 15]. The main distinction between this model and the symmetric interdiction model is that both the interdictor and the optimizer in our problem are constrained by the same feasibility conditions, whereas the interdictor was constrained by a budget on the number of edges in previous work.

The SMI problem is similar to the matching interdiction problem studied by Kamalian *et al.* [14, 13], the key difference being that the interdictor's matching is also required to be a maximum matching in their case. We show that this restriction can produce suboptimal SMI solutions; indeed, the results of [14, 13] have no implication for SMI. More broadly, interdiction problems have a long history, having been studied for military applications in the Cold War [20]. Closer to our work, they have been used to model competitive markets in economic theory. In particular, in the Stackelberg model [24], two firms compete sequentially on the quantity of output they produce of a homogeneous good. Furthermore, both players play by the same rules and therefore must operate under the same constraints. This is conceptually identical to our symmetric interdiction model and we hope that this model will be applied to other domains in the future.

2 Symmetric Interdiction: A General Framework

In this section, we give a general theorem that relates symmetric interdiction problems to their corresponding optimization problems for a broad class of optimization problems called *packing* problems. This includes many classical problems such as maximum matching, knapsack, maximum flow, etc. Formally, packing problems are those that can be encoded by the linear program (LP) given below, where all entries of the coefficient matrix $\mathbf{A}$, and that of vectors $\mathbf{b}$ and $\mathbf{c}$ are non-negative:

$$\text{maximize}\quad \mathbf{c}^\intercal \mathbf{x}, \quad \text{subject to}\quad \mathbf{A}\mathbf{x} \le \mathbf{b} \text{ and } \mathbf{0} \le \mathbf{x} \le \mathbf{1}. \tag{1}$$

Suppose $\mathbf{x}$ is a feasible solution to LP (1). Then, we define the *residual LP* of $\mathbf{x}$ as:

$$\text{maximize}\quad \mathbf{c}^\intercal \mathbf{y}, \quad \text{subject to}\quad \mathbf{A}\mathbf{y} \le \mathbf{b} \text{ and } \mathbf{0} \le \mathbf{y} \le \mathbf{1} - \mathbf{x}. \tag{2}$$

The *symmetric interdiction problem* is to find a feasible solution $\mathbf{x}$ that minimizes the optimal solution to the residual LP of $\mathbf{x}$. While we only focus on packing problems in this paper, we note that one can analogously define symmetric interdiction for covering problems[1] as well. Additionally, we note that all results in this section hold when $\mathbf{x}$ and $\mathbf{y}$ are constrained to be integral.

We call LP (1) the *optimization problem.* In this section, we develop a framework to obtain approximate solutions to the interdiction problem using exact/approximate solutions to the optimization problem. Before stating the result formally (Theorem 3), we set up some basic notation. Let

$$\mathbf{x} \setminus \mathbf{x}' = \begin{pmatrix} \max(0, x_1 - x'_1) \\ \vdots \\ \max(0, x_n - x'_n) \end{pmatrix}. \tag{3}$$

Note that $\mathbf{x} \setminus \mathbf{x}'$ is feasible if $\mathbf{x}$ and $\mathbf{x}'$ are feasible.

Let $\mathbf{x}^*$ be an optimal solution to the interdiction problem, and let $\mathbf{y}^*$ be an optimal solution to the residual LP w.r.t. $\mathbf{x}^*$. Now, consider a solution $\mathbf{x}$ that is feasible for LP (1). Ideally, we would like to claim that if $\mathbf{x}$ is an approximately optimal solution for the optimization problem, then it is also an approximately optimal for the interdiction problem. Unfortunately, this may not be true in general. However, we can show this connection between optimization and interdiction if $\mathbf{x}$ satisfies the following stronger condition:

$$\mathbf{c}^\intercal(\mathbf{x}^* \setminus \mathbf{x}) \le \alpha \cdot \mathbf{c}^\intercal(\mathbf{x} \setminus \mathbf{x}^*) \text{ for some approximation factor } \alpha \ge 1. \tag{4}$$

This condition says that after removing any overlap between the interdiction and optimization solutions, the approximation ratio must be α. For example, consider an optimal interdiction solution M^* to the maximum matching problem, and another matching M. After removing edges that appear in both M and M^*, the number of remaining edges in M must be within a factor α of the number of remaining edges in M^*. In particular, when $\mathbf{x}$ is an optimal solution to the optimization problem, condition (4) holds with $\alpha = 1$ for any maximization problem. Now, we formally state and prove the theorem that establishes the relationship between optimization and interdiction.

▶ **Theorem 3.** *Let $\mathbf{x}^*$ be an optimal solution to the interdiction problem, and let $\mathbf{y}^*$ be an optimal solution to the residual LP w.r.t. $\mathbf{x}^*$. Suppose $\mathbf{x}$ is a feasible solution satisfying condition (4), i.e., $\mathbf{c}^\intercal(\mathbf{x}^* \setminus \mathbf{x}) \le \alpha \cdot \mathbf{c}^\intercal(\mathbf{x} \setminus \mathbf{x}^*)$. Then, $\mathbf{x}$ is a $(1+\alpha)$-approximation to the corresponding interdiction problem. That is, if $\mathbf{y}$ is an optimal solution to the residual LP of $\mathbf{x}$, then $\mathbf{c}^\intercal \mathbf{y} \le (1+\alpha) \cdot \mathbf{c}^\intercal \mathbf{y}^*$.*

[1] Covering problems are minimization problems where the constraints are $\mathbf{A}\mathbf{x} \ge \mathbf{b}$, with the same non-negativity restrictions.

Proof. We define the intersection $\mathbf{x} \cap \mathbf{x}'$ to be

$$\mathbf{x} \cap \mathbf{x}' = \begin{pmatrix} \min(x_1, x_1') \\ \vdots \\ \min(x_n, x_n') \end{pmatrix}.$$

Observe that $\mathbf{c}^\intercal \cdot \mathbf{y} = \mathbf{c}^\intercal \cdot (\mathbf{y} \setminus (\mathbf{1} - \mathbf{x}^*)) + \mathbf{c}^T \cdot (\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*))$. We upper bound each summand of this equation. We will need to use the fact that $\mathbf{x} \setminus \mathbf{x}^*$ is a feasible solution to the residual LP of $\mathbf{x}^*$. This follows from two observations: (1) $\mathbf{x}$ satisfies the constraint $\mathbf{Ax} \leq \mathbf{b}$ which implies $\mathbf{x} \setminus \mathbf{x}^*$ does too, and (2) $\mathbf{x} \leq \mathbf{1}$ implies $\mathbf{x} \setminus \mathbf{x}^* \leq \mathbf{1} - \mathbf{x}^*$. We first upper bound the left summand:

$$\begin{aligned} \mathbf{c}^\intercal(\mathbf{y} \setminus (\mathbf{1} - \mathbf{x}^*)) &\leq \mathbf{c}^\intercal((\mathbf{1} - \mathbf{x}) \setminus (\mathbf{1} - \mathbf{x}^*)) && \text{(since } \mathbf{y} \text{ is feasible for the residual LP of } \mathbf{x}) \\ &\leq \mathbf{c}^\intercal(\mathbf{x}^* \setminus \mathbf{x}) \\ &\leq \alpha \cdot \mathbf{c}^\intercal(\mathbf{x} \setminus \mathbf{x}^*) && \text{(by assumption that } \mathbf{x} \text{ satisfies (4))} \\ &\leq \alpha \cdot \mathbf{c}^\intercal \cdot \mathbf{y}^* \\ & \qquad \text{(since } \mathbf{x} \setminus \mathbf{x}^* \text{ is feasible for the residual LP of } \mathbf{x}^*\text{, shown above)} \end{aligned}$$

Next we bound the right summand. Note that $\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*)$ is feasible for the residual LP of $\mathbf{x}^*$ since $\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*) \leq (\mathbf{1} - \mathbf{x}^*)$. Therefore, since $\mathbf{y}^*$ is optimal for the residual LP of $\mathbf{x}^*$, we have $\mathbf{c}^\intercal \cdot (\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*)) \leq \mathbf{c}^\intercal \cdot \mathbf{y}^*$. Putting together the bounds on the left and right summands, we get

$$\mathbf{c}^\intercal \cdot \mathbf{y} = \mathbf{c}^\intercal \cdot (\mathbf{y} \setminus (\mathbf{1} - \mathbf{x}^*)) + \mathbf{c}^\intercal \cdot (\mathbf{y} \cap (\mathbf{1} - \mathbf{x}^*)) \leq \alpha \cdot \mathbf{c}^\intercal \cdot \mathbf{y}^* + \mathbf{c}^\intercal \cdot \mathbf{y}^* = (1 + \alpha) \cdot \mathbf{c}^\intercal \cdot \mathbf{y}^*. \blacktriangleleft$$

▶ **Corollary 4.** *Any optimal solution $\hat{\mathbf{x}}$ to the optimization problem, is a 2-approximation to the corresponding symmetric interdiction problem.*

Proof. Note that $\hat{\mathbf{x}} = (\hat{\mathbf{x}} \setminus \mathbf{x}^*) + (\hat{\mathbf{x}} \cap \mathbf{x}^*)$. Similarly, $\mathbf{x}^* = (\mathbf{x}^* \setminus \hat{\mathbf{x}}) + (\hat{\mathbf{x}} \cap \mathbf{x}^*)$. Since $\hat{\mathbf{x}}$ is an optimal solution to the optimization problem, $\mathbf{c}^\intercal \mathbf{x}^* \leq \mathbf{c}^\intercal \hat{\mathbf{x}}$. Therefore, $\mathbf{c}^\intercal(\mathbf{x}^* \setminus \hat{\mathbf{x}}) \leq \mathbf{c}^\intercal(\hat{\mathbf{x}} \setminus \mathbf{x}^*)$. ◀

3 Symmetric Matching Interdiction: A 3/2 Approximation

Let $G = (V, E)$ be a graph. Then the symmetric matching interdiction (SMI) problem is to find some matching M^* such that the maximum matching in the graph $(V, E \setminus M^*)$ is minimized.

From Corollary 4, we get that any maximum matching is a 2-approximation for the SMI problem. In fact, any *maximal* matching is also a 2-approximation.

▶ **Lemma 5.** *Any maximal matching is a 2-approximation for the symmetric matching interdiction problem.*

Proof. For a graph G, let M be a maximal matching and L be the maximum matching on $G \setminus M$. Each component of $M \cup L$ is a path or a cycle of alternating edges of M and L. Any edge that appears by itself in a component of $M \cup L$ must be in M, by the maximality of M.

Let C be a component of $M \cup L$ that contains at least one edge of L. We show that for any matching M^* on C, the maximum matching on $C \setminus M^*$ has at least $|L \cap C|/2$ edges, which will complete the proof. Let j be the number of edges of C. Then, $|L| = \frac{j}{2}$ if j is even, and $|L| \leq \frac{j+1}{2}$ if j is odd. That is, $|L| \leq \lceil \frac{j}{2} \rceil$.

We will show later by a case analysis in Lemma 8 that the maximum matching on $C \setminus M^*$ has at least $\lceil \frac{j-1}{3} \rceil$ edges for any M^*. Since $\lceil \frac{j}{2} \rceil / \lceil \frac{j-1}{3} \rceil \leq 2$ for integers $j \geq 2$, the lemma follows. ◀

This is better than the 3-approximation guarantee for maximal matchings that we get from Theorem 3. In fact, the approximation factor of 2 is the best achievable, if we were to choose an arbitrary maximum or maximal matching. Consider a length-4 path. The optimal interdiction solution contains the edges at the two ends, leaving a matching of size 1. On the other hand, the first and third edges form a maximum matching, but leaves behind a matching of size 2.

But, what if we choose the *best* maximum matching instead of an arbitrary one? In the previous example, the optimal interdiction solution also turned out to be a maximum matching. Our first result in this section is to show that there always exists a maximum matching that is a 3/2-approximation to the optimal interdiction matching. In the second part of this section, we make this result constructive, i.e., give a polynomial-time algorithm for finding such a maximum matching. Before describing our result, we note that the approximation factor of 3/2 is the best we can hope for from a maximum matching, even the best one. Consider a cycle of length 6. The optimal interdiction solution contains any pair of opposite edges, leaving behind two disjoint length-2 paths containing a matching of size 2. On the other hand, any maximum matching contains 3 edges, which leaves behind 3 disjoint components forming a matching of size 3.

3.1 Approximating the SMI problem with maximum matchings

We show that the maximum matching with the largest intersection with any fixed optimal solution to the SMI problem is a 3/2 approximation to the SMI problem. In this section, M^* denotes an optimal solution to SMI, i.e., a matching that minimizes the size of the maximum matching L^* in the remaining graph $(V, E \setminus M^*)$. M denotes a maximum matching on G, and L denotes a maximum matching in the remaining graph $(V, E \setminus M)$. All matchings and connected components that we refer to in this section are defined as sets of edges; hence, set operations are only on the edges and do not affect vertices.

For any M and L, the size of a matching on $(M \cup L) \setminus M^*$ serves as a lower bound on the size of L^*, since $(M \cup L) \subseteq E$. So, our goal will be to show that the size of L is at most 3/2 times the size of a matching that we construct in $(M \cup L) \setminus M^*$. We will show this individually for every component of $M \cup L$. Let C be a component of $M \cup L$. We say M is *locally* 3/2-competitive on C with respect to M^* if $C \setminus M^*$ contains a matching of at least 2/3 times the size of $C \cap L$. If M is locally 3/2-competitive for each component, then that implies an approximation factor of 3/2 overall.

For some fixed M^*, there are only certain types of components of $M \cup L$ that may not be locally competitive. We call these components *critical*, and define their structure below. Note that $M \cup L$ is a set of vertex disjoint paths and even-length cycles, since it is composed of two matchings.

▶ **Definition 6.** We call component C *critical* w.r.t. matching M^* if all the following hold:
1. C is an even-length path,
2. the edges at the two ends of C are in M^*, and
3. $C \setminus M^*$ is a set of length-2 paths.

We will show in Lemma 8 that critical components, as defined in Definition 6, are the only ones that may not be locally competitive. From Definition 6, for a component to be critical, it must be a path with ℓ edges, where $\ell \equiv 4 \mod 6$ edges. We call these components *bad*:

▶ **Definition 7.** Let C be a component of $M \cup L$. Call C *bad* if C is a path and $|C| \equiv 4 \mod 6$, where $|C|$ denotes the number of edges in C.

Note that all critical components are bad, but not vice-versa, since criticality also depends on the structure of M^*.

We next show that M is locally 3/2-competitive on all components that are not critical. In fact, the lemma gives tighter bounds, which will be helpful in developing an algorithm later. Note that, till now, the only assumption we have made about M is that it is a maximum matching, i.e., the next lemma holds for *all* maximum matchings.

▶ **Lemma 8.** *Fix M, L, M^*, and L^*. Let C be a component of $M \cup L$. Let ℓ^* denote the size of a maximum matching on $C \setminus M^*$, and c denote the number of edges in C. (Note that ℓ^*, summed over all components C, lower bounds the size of L^*.) Then,*

1. *If C is not bad and c is odd, $\ell^* \geq \frac{c-1}{3}$.*
2. *If C is not bad and c is even, $\ell^* \geq \frac{c}{3}$.*
3. *If C is bad but not critical, $\ell^* \geq \frac{c+2}{3}$.*
4. *If C is bad and critical, $\ell^* \geq \frac{c-1}{3}$.*

Proof. We find these lower bounds on ℓ^* by constructing a matching $\widehat{L}$ on $C \setminus M^*$. Note that $C \setminus M^*$ is either an even cycle or a set of vertex-disjoint paths. In the former case, we pick every alternate edge on the cycle in $\widehat{L}$. In the latter case, for each path, we pick every alternate edge in $\widehat{L}$, including the two edges at the ends for odd length paths. Let m^* denote $|M^* \cap C|$. $\widehat{L}$ has the following properties:

1. $\widehat{L}$ contains at least $\lceil \frac{c-m^*}{2} \rceil$ edges.
2. For each component of $C \setminus M^*$, $\widehat{L}$ contains at least one edge.

Next, we show that these two properties are sufficient to prove that for each of the 4 cases in the statement of the lemma, the corresponding inequality holds.

Case (1). Note that C must be a path, since all cycles have even length in the union of two matchings. Therefore, property 2 ensures that $\widehat{L}$ has at least $m^* - 1$ edges. Along with property 1, this implies $\ell^* \geq |\widehat{L}| \geq \min(m^* - 1, \frac{c-m^*}{2})$. Optimizing over the possible values of m^* then gives us $\ell^* \geq \frac{c-1}{3}$.

Case (2). C is either a path or a cycle. We treat these cases differently.

1. When C is a cycle, property 2 implies that $\widehat{L}$ has at least m^* edges. Along with property 1, this implies $\ell^* \geq |\widehat{L}| \geq \min(m^*, \frac{c-m^*}{2})$. Optimizing over the possible values of m^* gives $\ell^* \geq \frac{c}{3}$.
2. When C is a path, property 2 implies that $\widehat{L}$ has at least $m^* - 1$ edges. Identical to case (1) above, we can now infer that $\ell^* \geq \frac{c-1}{3}$. Since ℓ^* is integral, we can claim that $\ell^* \geq \lceil \frac{c-1}{3} \rceil$. We also know that c is even and $c \not\equiv 4 \mod 6$. Together, this shows that $\lceil \frac{c-1}{3} \rceil \geq \frac{c}{3}$, which implies that $\ell^* \geq \frac{c}{3}$.

Case (3). We subdivide into two cases based on the size of M^*. Note that $c \equiv 4 \mod 6$; hence, $\frac{c+2}{3}$ is an integer.

1. Suppose $m^* \neq \frac{c+2}{3}$. If $m^* \geq \frac{c+2}{3} + 1$, then property (2) implies $\widehat{L} \geq \frac{c+2}{3}$. On the other hand, if $m^* \leq \frac{c+2}{3} - 1$, then property (1) ensures that $|\widehat{L}| \geq \lceil \frac{c+1/2}{3} \rceil = \frac{c+2}{3}$. In either case, $\ell^* \geq |\widehat{L}| \geq \frac{c+2}{3}$.
2. Suppose $m^* = \frac{c+2}{3}$. If M^* does not contain at least one end edge of path C, then $C \setminus M^*$ has m^* components, and therefore, property (2) ensures that $\widehat{L}$ has at least m^* edges. Now, consider the case where M^* contains both end edges of path C. In this case, the number of components in $C \setminus M^*$ is $m^* - 1 = \frac{c-1}{3}$. But, the total number of edges in

$C \setminus M^*$ is $c - m^* = \frac{2c-2}{3}$. Therefore, the average number of edges in each component of $C \setminus M^*$ is 2. Since C is not critical w.r.t. M^*, every component in $C \setminus M^*$ cannot have exactly 2 edges. As a consequence, there must be at least one component α in $C \setminus M^*$ that contains at least 3 edges. By property (2), $\widehat{L}$ contains at least $m^* - 1$ edges, but this matching can be augmented by picking a second edge from component α to produce a matching of size m^* in $C \setminus M^*$. Therefore, $\ell^* \geq m^* = \frac{c+2}{3}$.

Case (4). The proof is identical to the proof of case (1). ◀

We claim that the above lemma implies that M is locally 3/2-locally competitive on all non-critical components. Let ℓ denote the number of edges of L in C. In case (1), $\ell = \frac{c-1}{2}$, and in cases (2) and (3), $\ell = \frac{c}{2}$. Only case (4) is not locally 3/2-competitive, since $\ell = \frac{c}{2}$.

We now show that there is a maximum matching that has no critical components with respect to a fixed optimal M^*; this proves the existence of a 3/2-approximate maximum matching.

▶ **Lemma 9.** *A maximum matching with the largest intersection with some optimal solution M^* is a 3/2-approximation to the optimal interdiction solution, i.e., $|L| \leq \frac{3}{2}|L^*|$.*

Proof. Let M be a maximum matching with the largest intersection with M^* and let L and L^* be arbitrary maximum matchings in the respective remaining graphs. Let C be a critical component in $M \cup L$. Since $|C|$ is even, one of its end edges must be in L. Call this edge e, and let f denote its adjacent edge in C (note that f is in M). Since C is critical, we have $e \in M^*$. Then $(M \setminus \{f\}) \cup \{e\}$ is also a maximum matching. Since $e \in M^*$ and $f \notin M^*$, this contradicts the fact that we chose M as the maximum matching that maximizes $|M \cap M^*|$. ◀

3.2 A 3/2-Approximation algorithm

In this section, we make the results of the previous section constructive. If we knew M^*, we could give an algorithm that performed swaps of the kind used in the proof of Lemma 9. These swaps would each increase the size of $M \cap M^*$, and we would eventually obtain a solution with no critical components. Unfortunately, we don't know M^*. We show, however, that sometimes we can perform sets of swaps such that the overlap of M with *every* optimal solution M^* is increased. If such a set of swaps does not exist, we argue that our solution is already a 3/2-approximation.

The formal algorithm is given in Algorithm 1. We outline the steps here. We start with an arbitrary maximum matching M, and a maximum matching in $G \setminus M$. We then repeatedly perform swaps of the form given above on the set of all bad components for a total of $|E| + 1$ iterations. Finally, we output the best matching found over all these iterations. We argue that while a 3/2-approximate solution has not been obtained, each iteration of swaps increases the overlap of M with every optimal solution. Such an increase cannot happen more than $|E|$ times, and therefore a 3/2-approximate solution is found in some iteration of the algorithm.

▶ **Lemma 10.** *Let M be a maximum matching and L be a maximum matching in $G \setminus M$. Suppose there exists an optimal interdiction solution M^* such that M^* is critical on at most half the bad paths in $M \cup L$. Then, $|L| \leq \frac{3}{2}|L^*|$.*

Before proving Lemma 10, we show that this implies correctness of the algorithm. Suppose that for some iteration of the algorithm, the condition from Lemma 10 does not hold, i.e.,

Algorithm 1 A 3/2-approximation algorithm for the SMI problem.

```
1:  $M \leftarrow$ arbitrary maximum matching in $G$
2:  $L \leftarrow$ arbitrary maximum matching in $G \setminus M$
3:  $l_{\min} \leftarrow |L|$
4:  $M_{\min} \leftarrow M$
5:  for $j = 1 \rightarrow |E| + 1$ do
6:      if $|L| < l_{\min}$ then
7:          $l_{\min} \leftarrow |L|$
8:          $M_{\min} \leftarrow M$
9:      for bad path $C$ in $M \cup L$ do
10:         $M \leftarrow M \setminus \{e\} \cup \{f\}$   ▷ Let $e$ be the edge at the end of $C$ that is in $L$, $f \in M$ is
    the adjacent edge in $C$.
11:     $L \leftarrow$ arbitrary maximum matching in $G \setminus M$.
12: return $M_{\min}$
```

every optimal solution M^* is critical on strictly more than half the bad paths in $M \cup L$. After the for loop beginning on line 9, the size of the intersection between M and every optimal solution will have increased. This is because for every M^*, every $e \to f$ swap on a critical path increases the size of the overlap between M^* and M by 1, while every $e \to f$ swap on a non-critical bad path decreases the overlap by at most 1. This increase in overlap can happen at most $|E|$ times, so after $|E|+1$ iterations, we must have produced a solution M with $|L| \le \frac{3}{2}|L^*|$ as desired. We now prove Lemma 10 using Lemma 8. Although critical components have a local approximation ratio slightly worse than 3/2, non-critical bad paths offset this with a ratio better than 3/2.

Proof of Lemma 10. Let $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathcal{C}_4$ denote the sets of components of type (1), (2), (3), and (4) respectively from Lemma 8. Let ℓ^*_C denote a maximum matching on component $C \setminus M^*$. Also, let $E(C)$ denote the edges of component C and $E(\mathcal{C}_i) = \bigcup_{C \in \mathcal{C}_i} E(C)$. Then,

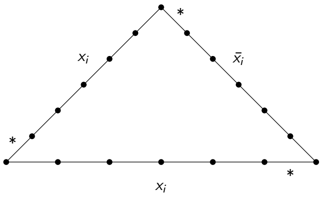
$$
\begin{aligned}
|L^*| &\ge \sum_{C \in \mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3 \cup \mathcal{C}_4} \ell^*_C \\
&\ge \sum_{C \in \mathcal{C}_1} \frac{|E(C)|-1}{3} + \sum_{C \in \mathcal{C}_2} \frac{|E(C)|}{3} + \sum_{C \in \mathcal{C}_3} \frac{|E(C)|+2}{3} + \sum_{C \in \mathcal{C}_4} \frac{|E(C)|-1}{3} \qquad \text{(from Lemma 8)} \\
&= \frac{|E(\mathcal{C}_1)| - |\mathcal{C}_1|}{3} + \frac{|E(\mathcal{C}_2)|}{3} + \frac{|E(\mathcal{C}_3)| + 2|\mathcal{C}_3|}{3} + \frac{|E(\mathcal{C}_4)| - |\mathcal{C}_4|}{3} \\
&\ge \frac{|E(\mathcal{C}_1)| - |\mathcal{C}_1|}{3} + \frac{|E(\mathcal{C}_2)|}{3} + \frac{|E(\mathcal{C}_3)| + |\mathcal{C}_3|}{3} + \frac{|E(\mathcal{C}_4)|}{3} \\
&\qquad \text{(since } |\mathcal{C}_3| \ge |\mathcal{C}_4|, \text{ i.e., at most half of all bad paths are critical)} \\
&= \frac{2}{3}|L \cap \mathcal{C}_1| + \frac{2}{3}|L \cap \mathcal{C}_2| + \frac{2|L \cap \mathcal{C}_3| + |\mathcal{C}_3|}{3} + \frac{2}{3}|L \cap \mathcal{C}_4| \ge \frac{2}{3}|L|. \\
&\qquad \text{(since } |L \cap C| = |C|/2 \text{ for } C \in \mathcal{C}_2 \cup \mathcal{C}_3 \cup \mathcal{C}_4 \text{ and } |L \cap C| = (|C|-1)/2 \text{ for } C \in \mathcal{C}_1\text{)} \qquad \blacktriangleleft
\end{aligned}
$$

4 Symmetric Matching Interdiction: Hardness of Approximation

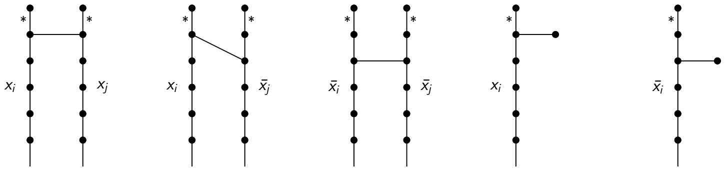
In this section, we show that the symmetric matching interdiction problem is APX-hard which rules out the possibility of a PTAS for the problem. We give an approximation-preserving reduction from a variant of MAX-SAT called 3-OCC-MAX-2-SAT that we define below.

▶ **Definition 11.** Let ϕ be a set of clauses, where each clause is a conjunction of at most 2 literals. Additionally, each variable appears in at most 3 literals in ϕ. Let k be an integer. (ϕ, k) is said to be in 3-OCC-MAX-2-SAT if there is a setting of the variables such that at least k clauses are satisfied.

3-OCC-MAX-2-SAT is known to be APX-hard [3]. To show the hardness of the SMI problem, we give an approximation preserving reduction from 3-OCC-MAX-2-SAT to the SMI problem. For the purposes of the reduction, we construct an instance graph G of the SMI problem from an instance of the 3-OCC-MAX-2-SAT problem (ϕ, k) as follows. For each variable x_i, we have a cycle in G containing $6z_i$ edges, where $z_i \leq 3$ is the number of times x_i appears as a literal in ϕ. We partition each cycle into z_i paths of length 6 each, which we call *literal paths*, such that each path is associated with one of the literals containing x_i. We order the edges of each path, denoting the first edge with '*' so that we can refer to the first, second, etc. edge on a literal path without ambiguity. The construction up until now is illustrated below:



We call all edges in such cycles *cycle edges*. Next, we add one edge to G for each clause in ϕ (we call these *clause edges*). Each clause contains either one or two literals. For a clause containing two literals, the clause edge connects the two literal paths corresponding to those literals. For a clause containing one literal, the clause edge connects that literal's path to a new vertex. Clause edges are adjacent to the second vertex on the literal path corresponding to a positive literal, and the third vertex on the literal path corresponding to a negative literal. Below, we illustrate the clauses $(x_i \vee x_j)$, $(x_i \vee \overline{x_j})$, $(\overline{x_i} \vee \overline{x_j})$, (x_i), and $(\overline{x_i})$, respectively.



This completes the construction of G. The following is our main technical lemma of the reduction.

▶ **Lemma 12.** *There is a setting of the variables that satisfies at least k clauses in ϕ if and only if there is a matching M such that in $G \setminus M$, the size of the maximum matching is at most $2\ell + m - k$, where m is the number of clauses in ϕ and ℓ is the number of literals.*

Before proving this lemma, we show that it is sufficient to prove APX-hardness of the SMI problem.

▶ **Theorem 13.** *Symmetric matching interdiction is APX-hard.*

Proof. Suppose we have an $(1+\epsilon)$-approximation to the SMI problem, i.e., a matching M in G such the maximum matching in $G \setminus M$ has size at most

$$(1+\epsilon)\cdot(2\ell + m - k) = 2\ell + m - \left[1 - \epsilon\left(\frac{2\ell+m}{k} - 1\right)\right]k.$$

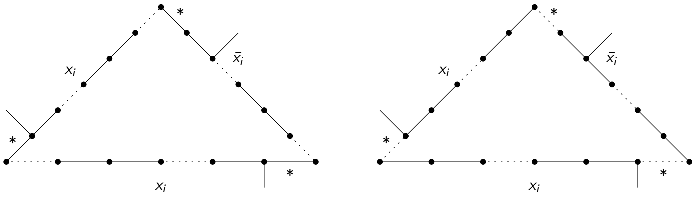
By Lemma 12, we can find a formula ϕ and an assignment $\mathbf{x}$ in the 3-OCC-MAX-2-SAT instance such that $\mathbf{x}$ satisfies at least $\left[1 - \epsilon\left(\frac{2\ell+m}{k} - 1\right)\right] k$ clauses of ϕ. Note that $\ell \leq 2m$ since each clause contains at most two literals; therefore, $2\ell + m \leq 5m$. If each variable is set i.i.d. to T/F with equal probability, then each clause is satisfied with probability 1/2 if it contains a single literal, and with probability 3/4 if it contains 2 literals. Therefore, the expected number of clauses satisfied by a 2-SAT formula under this random assignment is at least $m/2$. By the probabilistic method, it follows that the maximum number of satisfiable clauses $k \geq m/2$. Therefore, $m/k \leq 2$, which implies

$$1 - \epsilon((2\ell + m)/k - 1) \geq 1 - \epsilon(5m/k - 1) \geq 1 - 9\epsilon.$$

Therefore, this gives a $(1 - 9\epsilon)$-approximate solution to 3-OCC-MAX-2-SAT. ◀

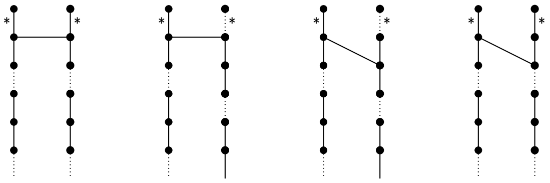
We spend the rest of the section proving Lemma 12. We first give a high level overview of the proof, and then give the technical details. We give a mapping from a setting of variables, $\mathbf{x}$ in ϕ, to a matching $M_{\mathbf{x}}$ in G. We argue that $\mathbf{x}$ satisfies k clauses of ϕ if and only if the maximum matching in $G \setminus M_{\mathbf{x}}$ contains $2\ell + m - k$ edges (Lemma 14). Then, we argue that for graph G produced by the reduction from a formula ϕ, there is a setting of variables $\mathbf{x}$ in ϕ such that $M_{\mathbf{x}}$ is the optimal solution to the SMI problem in G (Lemmas 15, 16, 17). Together, these lemmas prove Lemma 12.

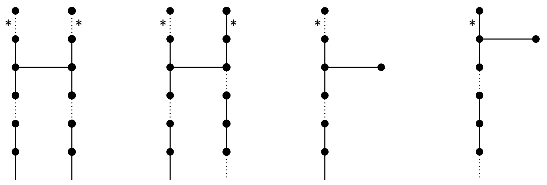
For an assignment $\mathbf{x}$ to the variables of ϕ, we construct matching $M_{\mathbf{x}}$ as follows: $M_{\mathbf{x}}$ does not contain any clause edge. $M_{\mathbf{x}}$ contains every third edge on each variable cycle. For the cycle corresponding to variable x_i, these edges are chosen in the following way: If x_i set to true, $M_{\mathbf{x}}$ contains the third and sixth edges of each literal path. We call $M_{\mathbf{x}}$ true on such a path. If x_i is set to false, $M_{\mathbf{x}}$ contains the first and fourth edges of each literal path. We call $M_{\mathbf{x}}$ false on such a path. For the cycle in G corresponding to variable x_i, we show the two possibilities for the edges in $G \setminus M_{\mathbf{x}}$ below; $M_{\mathbf{x}}$ is true on the first cycle, and false on the second.



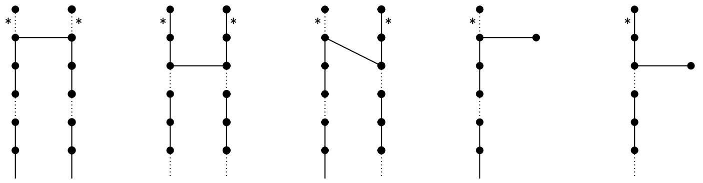
▶ **Lemma 14.** *An assignment $\mathbf{x}$ satisfies k clauses if and only if the maximum matching in $G \setminus M_{\mathbf{x}}$ has size $2\ell + m - k$.*

Proof. For each clause, we show that a maximum matching on the cycle and clause edges corresponding to that clause after removing $M_{\mathbf{x}}$ contains two edges for each literal in the clause, along with an additional edge if the clause is not satisfied by $\mathbf{x}$. This shows that the maximum matching on $G \setminus M_{\mathbf{x}}$ has size at most $2\ell + m - k$. Moreover, in each case there is a maximum matching that does not use the last edge on each literal path. Therefore, they can be combined into a single matching with $2\ell + m - k$ edges. To prove this, we enumerate over all types of clauses. Edges in $M_{\mathbf{x}}$ are drawn as dotted lines. The following are all possible satisfied clauses.





The following are the unsatisfied clauses:



This completes the proof. ◀

Lemma 14 implies the forward direction of Lemma 12. To show that the reduction holds in the other direction, we show that given any optimal solution M to the SMI problem, we can transform it to a matching $M_{\mathbf{x}}$ that is also optimal and corresponds to an assignment $\mathbf{x}$ of ϕ. We call such matchings that correspond to assignments in ϕ *consistent* matchings. The following are necessary and sufficient conditions for a matching to be consistent:

(a) On each variable cycle, the matching is either true or false (i.e. it contains either the third and sixth edges of each literal path, or the first and fourth edges), and

(b) the matching does not contain any clause edge.

We show that M can be transformed into a consistent matching as follows.

- If property (a) is violated, iteratively identify a cycle C on which M violates (a) and locally replace M with $M_{\mathbf{x}}$, which is defined below.
- Once only property (b) is violated, remove all remaining clause edges.

We show that neither of these steps increases the size of the maximum matching in $G \setminus M$; therefore, the eventual consistent matching is also optimal for the SMI problem.

First, we consider violations of property (a). Let assignment $\mathbf{x}$ be defined as follows: for each variable $x_i \in \mathbf{x}$, $x_i = true$ if x_i appears as at most one negative literal in ϕ and $x_i = false$ if x_i appears as at most one positive literal in ϕ. If M is not consistent, we will show that we can iteratively replace variable cycles of matching M with the corresponding variable cycles of $M_{\mathbf{x}}$.

M must violate property (a) on cycle C in one of the following two ways:

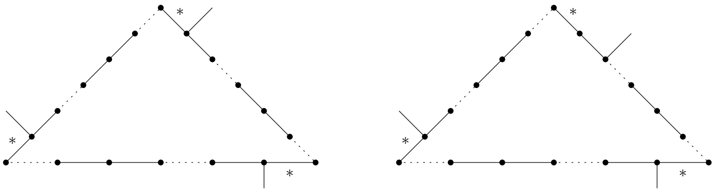
1. M does not contain every third edge of C.
2. M contains every third edge of C, but is neither true nor false on C (i.e. it contains the second and fifth edges of each literal path).

Let C_{clause} denote the set of clause edges adjacent to C. We will replace $M \cap (C \cup C_{clause})$ with $M_{\mathbf{x}} \cap C$ for violation (1), and $M \cap C$ with $M_{\mathbf{x}} \cap C$ for violation (2). We show in Lemmas 15 and 16 respectively that both these replacements result in valid matchings, and neither increases the size of the maximum matching in $G \setminus M$. Let $\xi_G(M)$ denote the size of the maximum matching in $G \setminus M$, and $\overline{C}$ denote $G \setminus (C \cup C_{clause})$.

▶ **Lemma 15.** *Consider a variable cycle C such that M does not contain every third edge of C. Then replacing M with $M' = (M \cap \overline{C}) \cup (M_{\mathbf{x}} \cap C)$ produces a matching, and does not increase the size of the maximum matching in $G \setminus M$.*

Proof. First, note that M' is a valid matching, since edges in $M \cap \overline{C}$ share no vertices with edges in $M_{\mathbf{x}} \cap C$. To complete the proof, we will show that $\xi_G(M') \leq \xi_G(M)$.

First, we claim that $\xi_C(M') \leq 3j+1 \leq \xi_C(M)$. The proof that $\xi_C(M) \geq 3j+1$ is very similar to the proof of case (3) of Lemma 8 and is not repeated here. The proof that $\xi_{C \cup C_{clause}}(M_{\mathbf{x}}) \leq 3j+1$ is by enumeration over all possible structures of $(C \cup C_{clause}) \cap M_{\mathbf{x}}$. We show two cases below. On the left, the variable x_C corresponding to the clause C appears as three true literals. On the right, it appears as two true literals and a false literal.



The maximum matching in the first graph has size $j/3 = 6$, and matching in the second has size $j/3+1 = 7$. It is straightforward to verify the other cases.

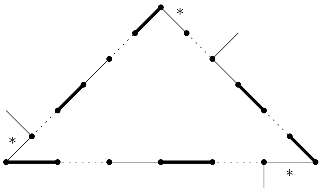
The rest of the proofs follows:

$$\begin{aligned}\xi_G(M) &\geq \xi_{\overline{C}}(M) + \xi_C(M) && \text{(vertex sets of } \overline{C} \text{ and } C \text{ are disjoint)}\\ &\geq \xi_{\overline{C}}(M) + \xi_{C\cup C_{clause}}(M_{\mathbf{x}})\\ &\geq \xi_G(M').\end{aligned}$$ ◀

▶ **Lemma 16.** *Consider a variable cycle C such that M contains every third edge of C, but is neither true nor false on C (i.e. M contains the second and fifth edge of each literal path). Then, replacing M with $M' = (M \cap (\overline{C} \cup C_{clause})) \cup (M_{\mathbf{x}} \cap C)$ produces a matching, and does not increase the size of the maximum matching in $G \setminus M$.*

Proof. It is not immediately clear that M' is a valid matching. To show that it is, it is sufficient to show that M does not contain any edges of C_{clause}. This follows from the fact that every edge of C_{clause} is adjacent to an edge of $M \cap C$ since M contains the second and fifth edges of each literal path of C. To complete the proof, we will show that $\xi_G(M') \leq \xi_G(M)$.

First, we claim $\xi_G(M) \geq \xi_{\overline{C} \cup C_{clause}}(M) + \xi_C(M)$. For this, it is enough to show that there is a matching on $C \setminus M$ of size $\xi_C(M)$ that leaves every vertex adjacent to a clause edge unmatched. The proof is by enumeration. We show one case below, it is straightforward to show the others. Edges of the matching on $C \setminus M$ are highlighted, and edges of M are shown as dotted lines.



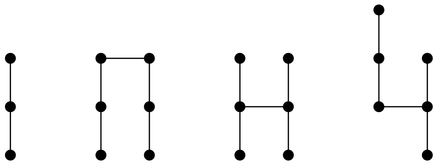
We can now complete the proof:

$$\begin{aligned}\xi_G(M) &\geq \xi_{\overline{C}\cup C_{clause}}(M) + \xi_C(M)\\ &= \xi_{\overline{C}\cup C_{clause}}(M) + \xi_C(M_{\mathbf{x}})\\ &\qquad (M \setminus C \text{ and } M_{\mathbf{x}} \setminus C \text{ are the same up to a rotation of cycle } C)\\ &\geq \xi_G(M').\end{aligned}$$ ◀

So, we can always replace M locally with $M_{\mathbf{x}}$ in a way that does not increase the size of the maximum matching in $G \setminus M$. By iteratively performing these replacements, we obtain a matching M which violates only property (b) of the consistency conditions. We now show that if matching M only violates property (b), any clause edge of M can be removed without changing the size of the maximum matching in $G \setminus M$.

▶ **Lemma 17.** *Suppose M is either true or false on all cycle edges, but M contains one or more clause edges. Then, removing the clause edges from M does not increase the size of the maximum matching in $G \setminus M$.*

Proof. $G \setminus M$ consists of a set of connected components, each of which is either a pair of cycle edges, or two pairs of cycle edges connected by a clause edge (either a path, a barbell, or a T as shown below):



Removing a clause edge from M transforms a pair of two-edge paths into a barbell in $G \setminus M$, which does not increase the size of the maximum matching. ◀

5 Randomized Symmetric Matching Interdiction

We now consider a randomized version of the symmetric matching interdiction problem. Rather than selecting matchings deterministically, the interdictor and the optimizer select *random* matchings M and L in G; the goal for the optimizer is to select M so as to minimize the maximum expected size of $L \setminus M$, the maximum taken over all choices of the random matching L. Note that unlike in the standard (deterministic) SMI model, the randomly chosen matchings M and L need not be disjoint. It is easy to see that L can be a (deterministically chosen) best response matching since the support of a randomized best response must consist only of best response matchings. Thus, formally, the randomized SMI problem is to find a probability distribution $\mathcal{M}$ over matchings that minimizes

$$\max_{\text{matching } L} \mathbb{E}[|L \setminus M|], \tag{5}$$

where M is a random matching drawn from $\mathcal{M}$. Any distribution (convex combination) over integral matchings, M, can be viewed as a fractional matching, i.e., as a point in the matching polytope [21]. Let $\bar{x} = \langle x_e \rangle$ denote a point in the matching polytope with x_e the probability (equivalently the fractional weight) of choosing edge e. The expected size of matching L in $G \setminus M$ is $\sum_{e \in L}(1 - x_e)$. Minimizing Eqn. 5 is therefore equivalent to minimizing over the matching polytope, the maximum over all matchings L, $\sum_{e \in L}(1 - x_e)$. This gives rise to the following LP, where, $E(S)$ denotes the set of edges with both endpoints in S, and $\delta(v)$ denotes the set of edges adjacent to v.

$$
\begin{aligned}
&\min y \\
&\text{s.t. } y \geq \sum_{e \in L} (1 - x_e) && \forall \text{ matchings } L \in G \\
&\qquad \sum_{e \in E(S)} x_e \leq \frac{|S| - 1}{2} && \forall S \subset V,\ S \text{ odd} \\
&\qquad \sum_{e \in \delta(v)} x_e \leq 1 && \forall v \in G \\
&\qquad 0 \leq x_e \leq 1 && \forall e \in G
\end{aligned}
\tag{6}
$$

The constraints on the x_e variables ensure that $\bar{x} = \langle x_e \rangle$ lies in the matching polytope [21]. This LP has exponentially many constraints (the first two sets of constraints – matching constraints and odd set constraints), so we give a separation oracle, enabling it to be solved using the ellipsoid algorithm [11]. For the matching constraints let z be the value of the maximum matching in graph G with edge weights of $(1 - x_e)$. If $y \geq z$, then the solution is feasible. Otherwise, the constraint corresponding to the matching with value z is violated. And for the odd set constraints we use the Gomory-Hu based separation oracle given by Padberg and Rao [18].

Thus, by solving the above LP, we can obtain the point, $\bar{x}$, in the matching polytope. However, we need a representation of this point as a convex combination of (or, distribution over) integral matchings in order to determine the (polynomial-time) strategy of the interdictor. Such a representation is guaranteed by the following known lemma (e.g. see [11]). For completeness, we give a proof in Appendix A.

▶ **Lemma 18.** *Let* $\mathbf{x}$ *be a fractional matching.* $\mathbf{x}$ *can be written as the convex combination of polynomially many integral matchings, and these matchings and their weights can be found in polynomial time.*

Finally, we note that there can be a gap of 2 between the optimal randomized and deterministic matchings. Consider a length 2 path. The optimal deterministic matching is either edge, and this matching has value 1 (since it leaves a matching of size one). On the other hand, the randomized matching that assigns probability 1/2 to each edge has value 1/2: Regardless of which edge is chosen to be the second matching, the expected size is 1/2.

6 Other Problems: Acyclic Subgraph Interdiction

As discussed in Section 2, our symmetric interdiction framework can be applied to a diverse set of combinatorial optimization problems. For example, consider any downward closed set system such as acyclic forests, independent vectors in a vector space, and more generally matroids; we can ask how much the interdictor can reduce some measure of the residual set system (e.g., rank) by removing a subset and its elements from the family (we can pose similar questions for families of upward closed sets). We illustrate this idea with the *symmetric acyclic subgraph interdiction problem*. The goal is to determine an acyclic subgraph T of a given graph G so as to minimize the maximum-size acyclic subgraph of $G \setminus T$. Our general framework implies a 2-approximation for this interdiction problem.

▶ **Lemma 19.** *An arbitrary spanning tree on* G *is a 2-approximation to symmetric acyclic subgraph interdiction, and this bound is tight.*

The above lemma follows directly from Corollary 4. We provide a different, more direct proof of this lemma below. This proof enables us to derive an example for which the bound is tight; i.e., there exists a graph G and a spanning tree of G that is at least a 2-approximate solution for G.

Proof of Lemma 19. We start with an alternate proof that an arbitrary spanning tree is at most a 2-approximation. Let T^* be a minimal optimal solution, and T be an arbitrary spanning tree. (If G is not connected, we argue on each component separately.) Note that for $S \subseteq G$, the size of the largest set of acyclic edges in $G \setminus S$ is $n - c$, where c is the number of components in $G \setminus S$.

Let c^* be the number of components in $G \setminus T^*$ and c be the number of components in $G \setminus T$. We consider two cases.

Case 1: $c^* \leq n/2$. Then since $c \geq 1$, $(n-c)/(n-c^*) \leq 2$.

Case 2: $c^* = n/2 + k$. The c^* components of $G \setminus T^*$ form a partition of G, where all of the edges of T^* cross the partition (by minimality of T^*). T must span the components of $G \setminus T^*$, and therefore

$$|T^* \cap T| \geq c^* - 1 = n/2 + k - 1.$$

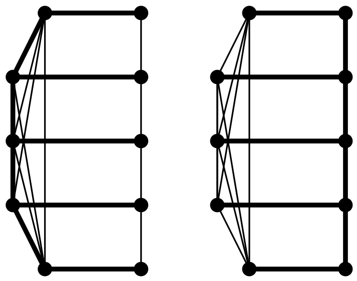
Additionally, $|T^*| \leq n - 1$, so we have

$$|T^* \setminus T| \leq n - 1 - n/2 - k + 1 = n/2 - k.$$

Starting from $G \setminus T^*$, adding back each edge of $T^* \setminus T$ can decrease the number of components by at most one. Therefore, the number of components of $(G \setminus T^*) \cup (T^* \setminus T) = G \setminus (T \cap T^*)$ is at least $(n/2 + k) - (n/2 - k) = 2k$. Therefore, the edges of T partition G into at least $2k$ components, i.e. $c \geq 2k$. Then we have

$$\frac{n-c}{n-c^*} = \frac{n-c}{n/2-k} \leq \frac{n-2k}{n/2-k} = 2. \qquad ◀$$

Next, we show that the bound is tight. Consider a graph with n vertices, such that $n/2$ vertices form a complete graph and $n/2$ vertices form a line. Additionally, there is an edge between the ith vertex on the line, and the ith vertex in the complete graph (vertices in the complete graph have arbitrary order). The optimal spanning tree is the line and all connecting edges, which leaves behind $\frac{n}{2} + 1$ components. A spanning tree that does not contain any edges of the line leaves just 2 components. The construction for $n = 10$ is shown below, with OPT on the right, and a bad solution on the left.



7 Concluding Remarks

In this paper, we have introduced a new symmetric interdiction model, and have focused on symmetric matching interdiction, for which we establish APX-hardness and a polynomial-time achievable 1.5-approximation. The symmetric interdiction model naturally extends to other matroid problems, as illustrated by the acyclic subgraph interdiction problem. Studying symmetric interdiction versions of other combinatorial optimization problems defined on matroids in an interesting direction of future research.

References

1 Douglas S. Altner, Özlem Ergun, and Nelson A. Uhan. The maximum flow network interdiction problem: valid inequalities, integrality gaps, and approximability. *Operations Research Letters*, 38(1):33–38, 2010.

2 Imre Bárány and Roman Karasev. Notes about the carathéodory number. *Discrete & Computational Geometry*, 48(3):783–792, 2012.

3 Piotr Berman and Marek Karpinski. On some tighter inapproximability results. In *International Colloquium on Automata, Languages, and Programming*, pages 200–209. Springer, 1999.

4 Stephen R. Chestnut and Rico Zenklusen. Interdicting structured combinatorial optimization problems with {0, 1}-objectives. *Mathematics of Operations Research*, 2016.

5 Stephen R. Chestnut and Rico Zenklusen. Hardness and approximation for network flow interdiction. *Networks*, 2017.

6 Richard L. Church, Maria P. Scaparra, and Richard S. Middleton. Identifying critical infrastructure: the median and covering facility interdiction problems. *Annals of the Association of American Geographers*, 94(3):491–502, 2004.

7 Eugene Peter Durbin. An interdiction model of highway transportation. *Rand Memorandum*, 1966.

8 Greg N. Frederickson and Roberto Solis-Oba. Increasing the weight of minimum spanning trees. *Journal of Algorithms*, 33(2):244–266, 1999.

9 P. M. Ghare, Douglas C. Montgomery, and W. C. Turner. Optimal interdiction policy for a flow network. *Naval Research Logistics Quarterly*, 18(1):37–45, 1971.

10 Bruce Golden. A problem in network interdiction. *Naval Research Logistics Quarterly*, 25(4):711–713, 1978.

11 Martin Grötschel, Lászlo Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, second corrected edition edition, 1993.

12 Alpár Jüttner. On budgeted optimization problems. *SIAM Journal on Discrete Mathematics*, 20(4):880–892, 2006.

13 Rafael R. Kamalian and Vahan V. Mkrtchyan. Two polynomial algorithms for special maximum matching constructing in trees. *arXiv preprint arXiv:0707.2295*, 2007.

14 Rafael R. Kamalian and Vahan V. Mkrtchyan. On complexity of special maximum matchings constructing. *Discrete Mathematics*, 308(10):1792–1800, 2008.

15 Leonid Khachiyan, Endre Boros, Konrad Borys, Khaled Elbassioni, Vladimir Gurvich, Gabor Rudolf, and Jihui Zhao. On short paths interdiction problems: Total and node-wise limited interdiction. *Theory of Computing Systems*, 43(2):204–233, 2008.

16 Churlzu Lim and J. Cole Smith. Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions*, 39(1):15–26, 2007.

17 Rui Miao, Rahul Potharaju, Minlan Yu, and Navendu Jain. The Dark menace: Characterizing network-based attacks in the cloud. In *Proceedings of the 2015 ACM Internet Measurement Conference*, 2015.

18 Manfred W. Padberg and M. R. Rao. Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7(1):67–80, 1982. doi:10.1287/moor.7.1.67.

19 Jörg Rambau. On a generalization of schönhardt's polyhedron. *Combinatorial and computational geometry*, 52:510–516, 2003.

20 Alexander Schrijver. On the history of the transportation and maximum flow problems. *Mathematical Programming*, 91(3):437–445, 2002.

21 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency. Vol. A. Paths, flows, matchings. Chapters 1–38.* Algorithms and combinatorics. Springer-Verlag, 2003. URL: http://opac.inria.fr/record=b1100334.

22 F Bruce Shepherd and Adrian Vetta. The demand-matching problem. *Mathematics of Operations Research*, 32(3):563–578, 2007.

23 Adrian Vetta and Gwenaël Joret. Reducing the rank of a matroid. *Discrete Mathematics & Theoretical Computer Science*, 17, 2015.

24 Heinrich Von Stackelberg. *Market structure and equilibrium.* Springer Science & Business Media, 2010.

25 R. Kevin Wood. Deterministic network interdiction. *Mathematical and Computer Modelling*, 17(2):1–18, 1993.

26 Rico Zenklusen. Matching interdiction. *Discrete Applied Mathematics*, 158(15):1676–1690, 2010.

27 Rico Zenklusen. Network flow interdiction on planar graphs. *Discrete Applied Mathematics*, 158(13):1441–1455, 2010.

28 Rico Zenklusen. An o(1)-approximation for minimum spanning tree interdiction. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 709–728. IEEE, 2015.

A Representation as a convex combination

We need to show that the representation of fractional matching as a convex combination of integral matchings can be obtained in polynomial-time. We show something more general – namely, that given a feasible point in the polytope defined by a set of linear constraints (even an exponential number in implicit form as a separation oracle [11]) we can find a representation of the point as a convex combination of the vertices of the polytope, in polynomial-time. Our constructive algorithm is folklore, but for completeness, we describe it in its entirety.

Let PT represent the d-dimensional polytope (convex bounded polyhedron) of solutions to $LP = \{C\}$, a set of linear constraints with F_C denoting the face, of dimension at most $d-1$, generated by constraint C. Let $\hat{p} \in PT$ be the given point. The set of constraints may be given in explicit form or implicitly with a bounding ball and a separation oracle [11]. The main idea is to take the ray starting at any vertex v of PT through $\hat{p}$ to its intersection p_i with the face opposite, F_C, then recursively represent p_i as the convex combination of vertices V_C of F_C; now it is an easy matter to see that $\hat{p}$ can be represented as a convex combination of $v \cup V_C$. In fact, it is easy to see, by strengthening the inductive hypothesis, that $\hat{p}$ is the convex combination of at most $d+1$ vertices of PT. All that is left to do is to see that a vertex of a polytope, the point of intersection of a ray with a face of the polytope and the representation of the face as a set of constraints (along with bounding ball and separation oracle, in the general case) can all be computed in polynomial-time. These operations are easy to compute when the constraints are given in explicit form and we leave it to the reader as an exercise. In the general setting of the separation oracle, a vertex can be computed using the ellipsoid algorithm [11]; the point of intersection of a ray with a face can be computed using binary search; and the polytope restricted to the face F_C can be represented by the same separation oracle augmented with the constraint that C be satisfied with equality, i.e the restricted polytope lies on the hyperplane (the bounding ball stays the same).

Note that the above proof shows that any point in a d-dimensional polytope lies in a simplex formed by at most $d + 1$ vertices of the polytope. This gives an alternate proof of Caratheodory's theorem [2]. It also shows that the simplex can be chosen to contain any particular vertex of the polytope. As a small digressional note, it is worth pointing out that the above technique does not extend to showing that every polyhedron (not polytope, polyhedrons may be non-convex) can be triangulated (simpliciated); in fact, though every polyhedron in 2 dimensions (i.e. polygon) can be triangulated this is not true in 3 (or higher) dimensions [19].

A Lottery Model for Center-Type Problems with Outliers*

David G. Harris[1], Thomas Pensyl[2], Aravind Srinivasan[3], and Khoa Trinh[4]

1 **Department of Computer Science, University of Maryland, College Park, USA**
davidgharris29@gmail.com

2 **Department of Computer Science, University of Maryland, College Park, USA**
tpensyl@cs.umd.edu

3 **Department of Computer Science and Institute for Advanced Computer Studies, University of Maryland, College Park, USA**
srin@cs.umd.edu

4 **Department of Computer Science, University of Maryland, College Park, USA**
khoa@cs.umd.edu

Abstract

In this paper, we give tight approximation algorithms for the k-center and matroid center problems with outliers. Unfairness arises naturally in this setting: certain clients could always be considered as outliers. To address this issue, we introduce a lottery model in which each client j is allowed to submit a parameter $p_j \in [0,1]$ and we look for a random solution that covers every client j with probability at least p_j. Out techniques include a randomized rounding procedure to round a point inside a matroid intersection polytope to a basis plus at most one extra item such that all marginal probabilities are preserved and such that a certain linear function of the variables does not decrease in the process with probability one.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.2.1 Combinatorics

Keywords and phrases approximation algorithms, randomized rounding, clustering problems

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.10

1 Introduction

The classic k-center and Knapsack Center problems are known to be approximable to within factors of 2 and 3 respectively [5]. These results are best possible unless P=NP [6, 5]. In these problems, we are given a metric graph G and want to find a subset $\mathcal{S}$ of vertices of G subject to either a cardinality constraint or a knapsack constraint such that the maximum distance from any vertex to the nearest vertex in $\mathcal{S}$ is as small as possible. We shall refer to vertices in G as *clients*. Vertices in $\mathcal{S}$ are also called *centers*.

It is not difficult to see that a few *outliers* (i.e., very distant clients) may result in a very large optimal radius in the center-type problems. This issue was raised by Charikar et. al. [2], who proposed a *robust* model in which we are given a parameter t and only need to serve t out of given n clients (i.e. $n-t$ *outliers* may be ignored in the solution). Here we consider three robust center-type problems: the Robust k-Center (RkCenter) problem, the Robust

* Research supported in part by NSF Awards CNS-1010789 and CCF-1422569, and by a research award from Adobe, Inc.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 10; pp. 10:1–10:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Knapsack Center (RKnapCenter) problem, and the Robust Matroid Center (RMatCenter) problem.

Formally, an instance $\mathcal{I}$ of the RkCenter problem consists of a set V of vertices, a metric distance d on V, an integer k, and an integer t. Let $n = |V|$ denote the number of vertices (clients). The goal is to choose a set $\mathcal{S} \subseteq V$ of centers (facilities) such that (i) $|\mathcal{S}| \leq k$, (ii) there is a set of *covered* vertices (clients) $\mathcal{C} \subseteq V$ of size at least t, and (iii) the objective function

$$R := \max_{j \in \mathcal{C}} \min_{i \in \mathcal{S}} d(i,j)$$

is minimized.

In the RKnapCenter problem, we are given a budget $B > 0$ instead of k. In addition, each vertex $i \in V$ has a weight $w_i \in \mathbb{R}_+$. The cardinality constraint (i) is replaced by the knapsack constraint: $\sum_{i \in \mathcal{S}} w_i \leq B$. Similarly, in the RMatCenter problem, the constraint (i) is replaced by a matroid constraint: $\mathcal{S}$ must be an independent set of a given matroid $\mathcal{M}$. Here we assume that we have access to the rank oracle of $\mathcal{M}$.

In [2], the authors introduced a greedy algorithm for the RkCenter problem that achieves an approximation ratio of 3. Recently, Chakrabarty et. al. [1] give a 2-approximation algorithm for this problem. Since the k-center problem is a special case of the RkCenter problem, this ratio is best possible unless P=NP.

The RKnapCenter problem was first studied by Chen et. al. [3]. In [3], the authors show that one can achieve an approximation ratio of 3 if allowed to slightly violate the knapsack constraint by a factor of $(1+\epsilon)$. It is still unknown whether there exists a true approximation algorithm for this problem. The current inapproximability bound is still 3 due to the hardness of the Knapsack Center problem.

The current best approximation guarantee for the RMatCenter problem is 7 by Chen et. al. [3]. This problem has a hardness result of $(3-\epsilon)$ via a reduction from the k-supplier problem.

From a practical viewpoint, unfairness arises inevitably in the robust model: some clients will always be considered as outliers and hence not covered within the guaranteed radius. To address this issue, we introduce a *lottery model* for these problems. The idea is to randomly pick a solution from a *public list* such that each client $j \in V$ is guaranteed to be covered with probability at least p_j, where $p_j \in [0,1]$ is the success rate requested by j. In practice, one possible way to determine these p_j's is based on the cost that the clients are willing to pay for their probability of being served. Also, observe that the special case when $p_j = 1$ for all $j \in V$ is equivalent to the standard model.

In this paper, we introduce new approximation algorithms for these problems under this model. (Note that this model has been used recently for the k-center and Knapsack Center problems (without outliers) in [4], which will appear soon on arXiv. All the techniques and problems in [4] are different.) We also propose improved approximation algorithms for the RkCenter problem and the RMatCenter problem.

1.1 The Lottery Model

In this subsection, we formally define our lottery model for the above-mentioned problems. First, the *Fair* Robust k-Center (FRkCenter) problem is formulated as follows. Besides the parameters V, d, k and t, each vertex $j \in V$ has a "target" probability $p_j \in [0,1]$. We are interested in the minimum radius R for which there exists a distribution $\mathcal{D}$ on subsets of V such that a set $\mathcal{S}$ drawn from $\mathcal{D}$ satisfies the following constraints:

Coverage constraint: $|\mathcal{C}| \geq t$ with probability one, where $\mathcal{C}$ is the set of all clients in V that are within radius R from some center $\mathcal{S}$,
Fairness constraint: $\Pr[j \in \mathcal{C}] \geq p_j$ for all $j \in V$, where $\mathcal{C}$ is as in the coverage constraint,
Cardinality constraint: $|\mathcal{S}| \leq k$ with probability one.

Here we aim for a polynomial-time, randomized algorithm that can sample from $\mathcal{D}$. Note that the RkCenter is a special of this variant in which all p_j's are set to be zero.

The *Fair* Robust Knapsack Center (FRKnapCenter) problem and *Fair* Robust Matroid Center (FRMatCenter) problem are defined similarly except that we replace the cardinality constraint by a knapsack constraint and a matroid constraint, respectively. More formally, in the FRKnapCenter problem, we are given a budget $B \in \mathbb{R}^+$ and each vertex i has a weight $w_i \in \mathbb{R}^+$. We require the total weight of centers in $\mathcal{S}$ to be at most B with probability one. Similarly, in the FRMatCenter problem, we are given a matroid $\mathcal{M}$ and we require the solution $\mathcal{S}$ to be an independent set of $\mathcal{M}$ with probability one.

1.2 Our contributions and techniques

First of all, we give tight approximation algorithms for the RkCenter and RMatCenter problems.

▶ **Theorem 1.** *There exist a 2-approximation algorithm for the RkCenter problem*[1] *and a 3-approximation algorithm for the RMatCenter problem.*

Our main results for the lottery model are summarized in the following theorems.

▶ **Theorem 2.** *For any given constant $\epsilon > 0$ and any instance $\mathcal{I} = (V, d, k, t, \vec{p})$ of the FRkCenter problem, there is a randomized polynomial-time algorithm $\mathcal{A}$ which can compute a random solution $\mathcal{S}$ such that*

- *$|\mathcal{S}| \leq k$ with probability one,*
- *$|\mathcal{C}| \geq (1-\epsilon)t$, where $\mathcal{C}$ is the set of all clients within radius $2R$ from some center in $\mathcal{S}$ and R is the optimal radius,*
- *$\Pr[j \in \mathcal{C}] \geq (1-\epsilon)p_j$ for all $j \in V$.*

▶ **Theorem 3.** *For any $\epsilon > 0$ and any instance $\mathcal{I} = (V, d, w, B, t, \vec{p})$ of the FRKnapCenter problem, there is a randomized polynomial-time algorithm $\mathcal{A}$ which can return random solution $\mathcal{S}$ such that*

- *$\sum_{i \in \mathcal{S}} w_i \leq (1+\epsilon)B$ with probability one,*
- *$|\mathcal{C}| \geq t$, where $\mathcal{C}$ is the set of vertices within distance $3R$ from some vertex in $\mathcal{S}$,*
- *$\Pr[j \in \mathcal{C}] \geq p_j$ for all $j \in V$.*

Finally, the FRMatCenter can be reduced to (randomly) rounding a point in a matroid intersection polytope. We design a randomized rounding algorithm which can output a pseudo solution, which consists of a basis plus one extra center. By using a preprocessing step and a configuration LP, we can satisfy the matroid constraint exactly (respectively, knapsack constraint) while slightly violating the coverage and fairness constraints in the FRMatCenter (respectively, FRKnapCenter) problem. We believe these techniques could be useful in other facility-location problems (e.g., the matroid median problem [7, 10]) as well.

[1] A 2-approximation algorithm has also been found independently by Chakrabarty et. al. [1], and in a private discussion between Marek Cygan and Samir Khuller. Our algorithm here is different from the algorithm in [1].

▶ **Theorem 4.** *For any given constant $\gamma > 0$ and any instance $\mathcal{I} = (V, d, \mathcal{M}, t, \vec{p})$ of the* FRMatCenter *(respectively,* FRKnapCenter*) problem, there is a randomized polynomial-time algorithm $\mathcal{A}$ which can return a random solution $\mathcal{S}$ such that*

- *$\mathcal{S}$ is a basis of $\mathcal{M}$ with probability one, (respectively, $w(\mathcal{S}) \leq B$ with probability one)*
- *$|\mathcal{C}| \geq t - \gamma^2 n$, where $\mathcal{C}$ is the set of vertices within distance $3R$ from some vertex in $\mathcal{S}$,*
- *there exists a set $T \subseteq V$ of size at least $(1-\gamma)n$, which is deterministic, such that $\Pr[j \in \mathcal{C}] \geq p_j - \gamma$ for all $j \in T$.*

1.3 Organization

The rest of this paper is organized as follows. In Section 2, we review some basic properties of matroids and discuss a filtering algorithm which is used in later algorithms. Then we develop approximation algorithms for the FRkCenter, FRKnapCenter, and FRMatCenter problems in the next three sections.

2 Preliminaries

2.1 Matroid polytopes

We first review a few basic facts about matroid polytopes. For any vector z and set S, we let $z(S)$ denote the sum $\sum_{i \in S} z_i$. Let $\mathcal{M}$ be any matroid on the ground set Ω and $r_{\mathcal{M}}$ be its rank function. The matroid base polytope of $\mathcal{M}$ is defined by

$$\mathcal{P}_{\mathcal{M}} := \left\{ x \in \mathbb{R}^{\Omega} : x(S) \leq r_{\mathcal{M}}(S) \quad \forall S \subseteq \Omega; \quad x(\Omega) = r_{\mathcal{M}}(\Omega); \quad x_i \geq 0 \quad \forall i \in \Omega \right\}.$$

▶ **Definition 5.** Suppose $Ax \leq b$ is a valid inequality of $\mathcal{P}_{\mathcal{M}}$. A face D of $\mathcal{P}_{\mathcal{M}}$ (corresponding to this valid inequality) is the set $D := \{x \in \mathcal{P}_{\mathcal{M}} : Ax = b\}$.

The following theorem gives a characterization for any face of $\mathcal{P}_{\mathcal{M}}$ (See, e.g., [9, 8]).

▶ **Theorem 6.** *Let D be any face of $\mathcal{P}_{\mathcal{M}}$. Then it can be characterized by*

$$D = \left\{ x \in \mathbb{R}^{\Omega} : x(S) = r_{\mathcal{M}}(S) \quad \forall S \in \mathcal{L}; \quad x_i = 0 \quad \forall i \in J; \quad x \in \mathcal{P}_{\mathcal{M}} \right\},$$

where $J \subseteq \Omega$ and $\mathcal{L}$ is a chain family of sets: $L_1 \subset L_2 \subset \ldots \subset L_m$. Moreover, it is sufficient to choose $\mathcal{L}$ as any maximal chain $L_1 \subset L_2 \subset \ldots \subset L_m$ such that $x(L_i) = r_{\mathcal{M}}(L_i)$ for all $i = 1, 2, \ldots, m$.

▶ **Proposition 7.** *Let $x \in \mathcal{P}_{\mathcal{M}}$ be any point and I be the set of all tight constraints of $\mathcal{P}_{\mathcal{M}}$ on x. Suppose D is the face with respect to I. Then one can compute a chain family $\mathcal{L}$ for D as in Theorem 6 in polynomial time.*

▶ **Corollary 8.** *Let D be any face of $\mathcal{P}_{\mathcal{M}}$. Then it can be characterized by*

$$D = \left\{ x \in \mathbb{R}^{\Omega} : x(S) = b_S \quad \forall S \in \mathcal{O}; \quad x_i = 0 \quad \forall i \in J; \quad x \in \mathcal{P}_{\mathcal{M}} \right\},$$

where $J \subseteq \Omega$ and $\mathcal{O}$ is a family of pairwise disjoint sets: $O_1, O_2, \ldots, O_m$, and $b_{O_1}, \ldots, b_{O_m}$ are some integer constants.

Algorithm 1 RFiltering (x, y)

```
$V' \leftarrow \emptyset$
for each cluster $F_j$ in decreasing order of $s_j = \sum_{i \in V: d(i,j) \le R} x_{ij}$ do
    if $F_j$ is unmarked then
        $V' \leftarrow V' \cup \{j\}$
        Set all unmarked clusters $F_k$ (including $F_j$ itself) s.t. $F_k \cap F_j \neq \emptyset$ as marked.
        Let $c_j$ be the number of marked clusters in this step.
$\vec{c} \leftarrow (c_j : j \in V')$
return $(V', \vec{c})$
```

2.2 Filtering algorithm

All algorithms in this paper are based on rounding an LP solution. In general, for each vertex $i \in V$, we have a variable $y_i \in [0, 1]$ which represents the probability that we want to pick i in our solution. (In the standard model, y_i is the "extent" that i is opened.) In addition, for each pair of $i, j \in V$, we have a variable $x_{ij} \in [0, 1]$ which represents the probability that j is connected to i.

Note that in all center-type problems, the optimal radius R is always the distance between two vertices. Therefore, we can always "guess" the value of R in $O(n^2)$ time. WLOG, we may assume that we know the correct value of R. For any $j \in V$, we let $F_j := \{i \in V : d(i, j) \le R \wedge x_{ij} > 0\}$ and $s_j := \sum_{i \in V: d(i,j) \le R} x_{ij}$. We shall refer to F_j as a cluster with cluster center j. Depending on a specific problem, we may have different constraints on x_{ij}'s and y_i's. In general, the following constraints are valid in most of the problems here:

$$\sum_{j \in V} \sum_{i \in V: d(i,j) \le R} x_{ij} \ge t, \tag{1}$$

$$\sum_{i \in V: d(i,j) \le R} x_{ij} \le 1, \quad \forall j \in V, \tag{2}$$

$$x_{ij} \le y_i, \quad \forall i, j \in V, \tag{3}$$

$$y_i, x_{ij} \ge 0, \quad \forall i, j \in V. \tag{4}$$

For the *fair* variants, we may also require that

$$\sum_{i \in V: d(i,j) \le R} x_{ij} \ge p_j, \quad \forall j \in V. \tag{5}$$

Constraint (1) says that at least t vertices should be covered. Constraint (2) ensures that each vertex is only connected to at most one center. Constraint (3) means vertex j can only connect to center i if it is open. Constraint (5) says that the total probability of j being connected should be at least p_j. By constraints (2) and (3), we have $y(F_j) \le 1$.

The first step of all algorithms in this paper is to use the following *filtering* algorithm to obtain a maximal collection of disjoint clusters. The algorithm will return the set V' of cluster centers of the chosen clusters. In the process, we also keep track of the number c_j of other clusters removed by F_j for each $j \in V'$.

3 The k-center problems with outliers

In this section, we first give a simple 2-approximation algorithm for the RkCenter problem. Then, we give an approximation algorithm for the FRkCenter problem, proving Theorem 2.

Algorithm 2 RkCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
2: $\mathcal{S} \leftarrow$ the top $k$ vertices $i \in V'$ with highest value of $c_i$.
3: return $\mathcal{S}$
```

3.1 The robust k-center problem

Suppose $\mathcal{I} = (V, d, k, t)$ is an instance the RkCenter problem with the optimal radius R. Consider the polytope $\mathcal{P}_{\mathsf{RkCenter}}$ containing points (x, y) satisfying constraints (1)–(4), and the cardinality constraint:

$$\sum_{i \in V} y_i \leq k. \tag{6}$$

Since R is the optimal radius, it is not difficult to check that $\mathcal{P}_{\mathsf{RkCenter}} \neq \emptyset$. Let us pick any fractional solution $(x, y) \in \mathcal{P}_{\mathsf{RkCenter}}$. The next step is to round (x, y) into an integral solution using the simple Algorithm 2.

Analysis. By construction, the algorithm returns a set $\mathcal{S}$ of k open centers. Note that, for each $i \in \mathcal{S}$, c_i is the number of distinct clients within radius $2R$ from i. Thus, it suffices to show that $\sum_{i \in \mathcal{S}} c_i \geq t$. By inequality (2), we have that $s_j \leq 1$ for all $j \in V'$. Thus,

$$\sum_{i \in V'} c_i s_i \geq \sum_{i \in V} s_i \geq t,$$

where the first inequality is due to the greedy choice of vertices in V' and the second inequality follows by (1). Now recall that the clusters whose centers in V' are pairwise disjoint. By constraint (6), we have

$$\sum_{i \in V'} s_i \leq \sum_{i \in V'} y(F_i) \leq \sum_{i \in V} y_i \leq k.$$

It follows by the choice of $\mathcal{S}$ that $\sum_{i \in \mathcal{S}} c_i \geq t$. This concludes the first part of Theorem 1.

3.2 The fair robust k-center problem

Assume $\mathcal{I} = (V, d, k, t, \vec{p})$ be an instance of the FRkCenter problem with the optimal radius R. Fix any $\epsilon > 0$. If $k \leq 2/\epsilon$, then we can generate all possible $O\left(n^{1/\epsilon}\right)$ solutions and then solve an LP to obtain the corresponding marginal probabilities. So the problem can be solved easily in this case. We will assume that $k \geq 2/\epsilon$ for the rest of this section. Consider the polytope $\mathcal{P}_{\mathsf{FRkCenter}}$ containing points (x, y) satisfying constraints (1)–(4), the fairness constraint (5), and the cardinality constraint (6). We now show that $\mathcal{P}_{\mathsf{FRkCenter}}$ is actually a valid relaxation polytope.

▶ **Proposition 9.** *We have that $\mathcal{P}_{\mathsf{FRkCenter}} \neq \emptyset$.*

Fix any small parameter $\epsilon > 0$. The description of our algorithm is shown in Algorithm 3.

Analysis. First, note that one can find such a vector δ in line 5 as the system of $\delta(V') = 0$ and $\vec{c} \cdot \delta = 0$ consists of two constraints and at least 3 variables (and hence is underdetermined.) By construction, at least one more fractional variable becomes rounded after each iteration.

Algorithm 3 FRkCenterRound (ϵ, x, y)

```
$(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
for each $j \in V'$ do
    $y'_j \leftarrow (1-\epsilon)\sum_{i \in F_j} x_{ij}$
while $y'$ still contains $\geq 3$ fractional values in $(0,1)$ do
    Let $\delta \in \mathbb{R}^{V'}, \delta \neq 0$ be such that $\delta_i = 0 \;\; \forall i \in V' : y'_i \in \{0,1\}$, $\delta(V') = 0$, and $\vec{c} \cdot \delta = 0$.
    Choose scaling factors $a, b > 0$ such that
        - $y' + a\delta \in [0,1]^{V'}$ and $y' - b\delta \in [0,1]^{V'}$
        - there is at least one new entry of $y' + a\delta$ which is equal to zero or one
        - there is at least one new entry of $y' - b\delta$ which is equal to zero or one
    With probability $\frac{b}{a+b}$, update $y' \leftarrow y' + a\delta$; else, update $y' \leftarrow y' - b\delta$.
return $\mathcal{S} = \{i \in V : y'_i > 0\}$.
```

Thus, the algorithm terminates after $O(n)$ rounds. Let $\mathcal{S}$ denote the (random) solution returned by FRkCenterRound and $\mathcal{C}$ be the set of all clients within radius $3R$ from some center in $\mathcal{S}$. Theorem 2 can be verified by the following propositions.

▶ **Proposition 10.** $|\mathcal{S}| \leq k$ *with probability one.*

▶ **Proposition 11.** $|\mathcal{C}| \geq (1-\epsilon)t$ *with probability one.*

▶ **Proposition 12.** $\Pr[j \in \mathcal{C}] \geq (1-\epsilon)p_j$ *for all* $j \in V$*.*

4 The Knapsack Center problems with outliers

We study the RKnapCenter and FRKnapCenter problems in this section. Recall that in these problems, each vertex has a weight and we want to make sure that the total weight of the chosen centers does not exceed a given budget B. We first give a 3-approximation algorithm for the RKnapCenter problem that slightly violates the knapsack constraint. Although this is not better than the known result by [3], both our algorithm and analysis here are more natural and simpler. It serves as a starting point for the next results. For the FRKnapCenter, we show that it is possible to satisfy the knapsack constraint exactly with small violations in the coverage and fairness constraints.

4.1 The robust knapsack center problem

Suppose $\mathcal{I} = (V, d, w, B, t)$ is an instance the RKnapCenter problem with the optimal radius R. Consider the polytope $\mathcal{P}_{\mathsf{RKnapCenter}}$ containing points (x, y) satisfying constraints (1)–(4), and the knapsack constraint:

$$\sum_{i \in V} w_i y_i \leq B. \tag{7}$$

Again, it is not difficult to check that $\mathcal{P}_{\mathsf{RKnapCenter}} \neq \emptyset$. Let us pick any fractional solution $(x, y) \in \mathcal{P}_{\mathsf{RKnapCenter}}$. See Algorithm 4 for the pseudo-approximation algorithm to round (x, y).

Analysis. We first claim that $\mathcal{P}' \neq \emptyset$ which implies that the extreme point Y of $\mathcal{P}'$ (in line 4) does exist. To see this, let $z_i := s_i$ for all $i \in V'$. Then we have

$$\sum_{i \in V'} c_i z_i = \sum_{i \in V'} c_i s_i \geq \sum_{i \in V} s_i \geq t.$$

Algorithm 4 RKnapCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
2: For each $i \in V'$, let $v_i \leftarrow \arg\min_{j \in F_i}\{w_j\}$ be the vertex with smallest weight in $F_i$
3: Let $\mathcal{P}' := \left\{z \in [0,1]^{V'} : \sum_{i \in V'} c_i z_i \geq t \quad \wedge \quad \sum_{i \in V'} w_{v_i} z_i \leq B\right\}$
4: Compute an extreme point $Y$ of $\mathcal{P}'$
5: return $\mathcal{S} = \{v_i : i \in V,\ Y_i > 0\}$
```

Also,

$$\begin{aligned}\sum_{i \in V'} w_{v_i} z_i &= \sum_{i \in V'} w_{v_i} s_i \\ &= \sum_{i \in V'} w_{v_i} \sum_{j \in F_i} x_{ji} \\ &\leq \sum_{i \in V'} w_{v_i} \sum_{j \in F_i} y_j \\ &\leq \sum_{i \in V'} \sum_{j \in F_i} w_j y_j \leq \sum_{i \in V} w_i y_i \leq B.\end{aligned}$$

All the inequalities follow from LP constraints and definitions of s_i, c_i, and v_i. Thus, $z \in \mathcal{P}'$, implying that $\mathcal{P}' \neq \emptyset$.

► **Proposition 13.** RKnapCenterRound *returns a solution $\mathcal{S}$ such that $w(\mathcal{S}) \leq B + 2w_{max}$ and $|\mathcal{C}| \geq t$, where $\mathcal{C}$ is the set of vertices within distance $3R$ from some vertex in $\mathcal{S}$ and w_{max} is the maximum weight of any vertex in V.*

4.2 The fair robust knapsack center problem

In this section, we will first consider a simple algorithm that only violates the knapsack constraint by two times the maximum weight of any vertex. Then using a configuration polytope to "condition" on the set of "big" vertices, we show that it is possible to either violate the budget by $(1 + \epsilon)$ or to preserve the knapsack constraint while slightly violating the coverage and fairness constraints.

4.2.1 Basic algorithm

Suppose $\mathcal{I} = (V, d, w, B, t, \vec{p})$ is an instance the FRKnapCenter problem with the optimal radius R. Consider the polytope $\mathcal{P}_{\textsf{FRKnapCenter}}$ containing points (x, y) satisfying constraints (1)–(4), the fairness constraint (5), and the knapsack constraint (7). The proof that $\mathcal{P}_{\textsf{FRKnapCenter}} \neq \emptyset$ is very similar to that of Proposition 9 and is omitted here.

The following algorithm is a randomized version of RKnapCenterRound.

Analysis. It is not hard to verify that $\mathcal{P}' \neq \emptyset$ (see the analysis in Section 4.1). This means that the decomposition at line 4 can be done.

► **Proposition 14.** *The algorithm* BasicFRKnapCenterRound *returns a random solution $\mathcal{S}$ such that $w(\mathcal{S}) \leq B + 2w_{max}$, $|\mathcal{C}| \geq t$, and* $\Pr[j \in \mathcal{C}] \geq p_j$ *for all $j \in V$, where $\mathcal{C}$ is the set of vertices within distance $3R$ from some vertex in $\mathcal{S}$ and w_{max} is the maximum weight of any vertex in V.*

Algorithm 5 BasicFRKnapCenterRound (x, y)

```
$(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
For each $i \in V'$ let $v_i := \arg\min_{j \in F_i}\{w_j\}$ be the vertex with smallest weight in $F_i$
Let $\mathcal{P}' := \left\{z \in [0,1]^{V'} : \sum_{i \in V'} c_i z_i \geq t \quad \wedge \quad \sum_{i \in V'} w_{v_i} z_i \leq B\right\}$
Let $z_i \leftarrow s_i$ for all $i \in V'$. Write $z$ as a convex combination of extreme points
    $z^{(1)}, \ldots, z^{(n+1)}$ of $\mathcal{P}'$:

        $z = p_1 z^{(1)} + \ldots + p_{n+1} z^{(n+1)}$,

    where $\sum_\ell p_\ell = 1$ and $p_\ell \geq 0$ for all $\ell \in [n+1]$.
Randomly choose $Y \leftarrow z_\ell$ with probability $p_\ell$.
return $\mathcal{S} = \{v_i : i \in V,\ Y_i > 0\}$
```

Algorithm 6 FRKnapCenterRound1 (x, y, q)

```
Randomly pick a set $U \in \mathcal{U}$ with probability $q_U$
Let $x'_{ij} \leftarrow x^U_{ij}/q_U$ and $y'_i \leftarrow \min\{y^U_i/q_U, 1\}$
return $\mathcal{S} =$ BasicRFKnapCenterRound $(x', y')$
```

4.2.2 An algorithm slightly violating the budget constraint

Fix a small parameter $\epsilon > 0$. A vertex i is said to be *big* iff $w_i > \epsilon B$. Then there can be at most $1/\epsilon$ big vertices in a solution. Let $\mathcal{U}$ denote the collection of all possible sets of big vertices. We have that $|\mathcal{U}| \leq n^{O(1/\epsilon)}$. Consider the *configuration* polytope $\mathcal{P}_{\mathsf{config1}}$ containing points (x, y, q) with the following constraints:

$$\begin{cases} \sum_{U \in \mathcal{U}} q_U = 1 & \\ \sum_{i \in V : d(i,j) \leq R} x^U_{ij} \leq q_U & \forall j \in V, U \in \mathcal{U} \\ \sum_{U \in \mathcal{U}} \sum_{i \in V : d(i,j) \leq R} x^U_{ij} \geq p_j & \forall j \in V \\ x^U_{ij} \leq y^U_i & \forall i, j \in V, U \in \mathcal{U} \\ \sum_{i \in V} w_i y^U_i \leq q_U B & \forall U \in \mathcal{U} \\ \sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} x^U_{ij} \geq q_U t & \\ y^U_i = 1 & \forall U \in \mathcal{U}, i \in U \\ y^U_i = 0 & \forall U \in \mathcal{U}, i \in V \setminus U, w_i > 1/\epsilon \\ x^U_{ij}, y^U_i, q_U \geq 0 & \forall i, j \in V, U \in \mathcal{U} \end{cases}$$

We first claim that $\mathcal{P}_{\mathsf{config1}}$ is a valid relaxation polytope for the problem.

▶ **Proposition 15.** *We have that $\mathcal{P}_{\mathit{config1}} \neq \emptyset$.*

Next, let us pick any $(x, y, q) \in \mathcal{P}_{\mathsf{config1}}$ and use the following algorithm to round it. We are now ready to prove Theorem 3.

Proof of Theorem 3. We will show that FRKnapCenterRound1 will return a solution $\mathcal{S}$ with properties in Theorem 3. Let $\mathcal{E}(U)$ denote the event that $U \in \mathcal{U}$ is picked in the algorithm. Note that (x', y') satisfies the following constraints:

$$\begin{aligned}\sum_{j\in V}\sum_{i\in V:d(i,j)\le R} x'_{ij} &\ge t,\\ \sum_{i\in V:d(i,j)\le R} x'_{ij} &\le 1, \quad \forall j\in V,\\ \sum_{i\in V:d(i,j)\le R} x'_{ij} &= \sum_{i\in V:d(i,j)\le R} x_{ij}/q_U, \quad \forall j\in V,\\ x'_{ij} &\le y'_i, \quad \forall i,j\in V,\\ \sum_{i\in V} w_i y'_i &\le B.\end{aligned}$$

Moreover, $y'_i = 1$ for all $i \in U$ and $y'_i = 0$ for all $i \in V \setminus U$ and $w_i > \epsilon B$. Thus, the two extra fractional vertices opened by BasicFRKnapCenterRound will have weight at most ϵB. By Proposition 14, we have $w(\mathcal{S}) \le B + 2\epsilon B = (1+2\epsilon)B$. Moreover, conditioned on U, we have

$$\Pr[j\in\mathcal{C}|\mathcal{E}(U)] \ge \sum_{i\in V:d(i,j)\le R} x'_{ij} = \sum_{i\in V:d(i,j)\le R} x_{ij}/q_U.$$

Thus, by definition of $\mathcal{P}_{\mathsf{config1}}$ and our construction of $\mathcal{S}$, we get

$$\begin{aligned}\Pr[j\in\mathcal{C}] &= \sum_{U\in\mathcal{U}} \Pr[j\in\mathcal{C}|\mathcal{E}(U)]\Pr[\mathcal{E}(U)]\\ &\ge \sum_{U\in\mathcal{U}}\sum_{i\in V:d(i,j)\le R} x_{ij}\\ &\ge p_j.\end{aligned}$$ ◀

4.2.3 An algorithm that satisfies the knapsack constraint exactly

Let $\epsilon > 0$ a small parameter to be determined. Let $\mathcal{U}$ denote the collection of all possible sets of verticies with size at most $\lceil 1/\epsilon \rceil$. We have that $|\mathcal{U}| \le n^{O(1/\epsilon)}$. Suppose R is the optimal radius to our instance. Given a set $U \in \mathcal{U}$, we say that vertex $j \in V$ is *blue* if there exists $i \in U$ such that $d(i,j) \le 3R$. Otherwise, vertex i is said to be *red*. For any $i \in V$, let $\mathrm{RBall}(i, U, R)$ denote the set of red vertices within radius $3R$ from i:

$$\mathrm{RBall}(i,U,R) := \{j\in V : (d(i,j)\le 3R \ \wedge\ \nexists k\in U : d(k,j)\le 3R)\}.$$

Consider the *configuration* polytope $\mathcal{P}_{\mathsf{config2}}$ containing points (x, y, q) with the following constraints:

$$\begin{cases}\sum_{U\in\mathcal{U}} q_U = 1 & \\ \sum_{i\in V:d(i,j)\le R} x^U_{ij} \le q_U & \forall j\in V, U\in\mathcal{U}\\ \sum_{U\in\mathcal{U}}\sum_{i\in V:d(i,j)\le R} x^U_{ij} \ge p_j & \forall j\in V\\ x^U_{ij}\le y^U_i & \forall i,j\in V, U\in\mathcal{U}\\ \sum_{i\in V} w_i y^U_i \le q_U B & \forall U\in\mathcal{U}\\ \sum_{j\in V}\sum_{i\in V:d(i,j)\le R} x^U_{ij} \ge q_U t & \\ y^U_i = 1 & \forall U\in\mathcal{U}, i\in U\\ y^U_i = 0 & \forall U\in\mathcal{U}, i\in V\setminus U, |\mathrm{RBall}(i,U,R)|\ge \epsilon n\\ x^U_{ij}, y^U_i, q_U \ge 0 & \forall i,j\in V, U\in\mathcal{U}\end{cases}$$

Algorithm 7 FRKnapCenterRound2 (x, y, q)

```
Randomly pick a set $U \in \mathcal{U}$ with probability $q_U$
Let $x'_{ij} \leftarrow x^U_{ij}/q_U$ and $y'_i \leftarrow \min\{y^U_i/q_U, 1\}$
$\mathcal{S}' \leftarrow$ BasicRFKnapCenterRound $(x', y')$
Let $i_1, i_2$ be vertices in $\mathcal{S}' \setminus U$ having largest weights.
return $\mathcal{S} = \mathcal{S}' \setminus \{i_1, i_2\}$
```

We first claim that $\mathcal{P}_{\mathsf{config2}}$ is a valid relaxation polytope for the problem.

▶ **Proposition 16.** *We have that $\mathcal{P}_{config2} \neq \emptyset$.*

Next, let us pick any $(x, y, q) \in \mathcal{P}_{\mathsf{config2}}$ and use the Algorithm 7 to round it.

Analysis. Let us fix any $\gamma > 0$ and set $\epsilon := \frac{\gamma^2}{2}$. Also, let $\mathcal{E}(U)$ denote the event that $U \in \mathcal{U}$ is picked in the algorithm. Again, observe that (x', y') satisfies the following inequalities:

$$\sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} x'_{ij} \geq t,$$

$$\sum_{i \in V : d(i,j) \leq R} x'_{ij} \leq 1, \quad \forall j \in V,$$

$$\sum_{i \in V : d(i,j) \leq R} x'_{ij} = \sum_{i \in V : d(i,j) \leq R} x_{ij}/q_U, \quad \forall j \in V,$$

$$x'_{ij} \leq y'_i, \quad \forall i, j \in V,$$

$$\sum_{i \in V} w_i y'_i \leq B.$$

Recall that the algorithm BasicFRKnapCenterRound will return a solution $\mathcal{S}'$ consisting of a set $\mathcal{S}''$ with $w(\mathcal{S}'') \leq B$ plus (at most) two extra "fractional" centers i^* and i^{**}. Moreover, we have $0 < y'_{i^*}, y'_{i^{**}} < 0$, which implies that $i^*, i^{**} \notin U$. Thus, by removing the two centers having highest weights in $\mathcal{S}' \setminus U$, we ensure that the total weight of $\mathcal{S}$ is within the given budget B with probability one.

Now we shall prove the coverage guarantee. By Proposition 14, $\mathcal{S}'$ covers at least t vertices within radius $3R$. If a vertex is blue, it can always be connected to some center in U; and hence, it is not affected by the removal of i_1, i_2. Because each of i_1 and i_2 can cover at most ϵn other red vertices, we have

$$|\mathcal{C}| \geq t - 2\epsilon n = 1 - \gamma^2 n.$$

For any $j \in V$, let X_j be the random indicator for the event that j is covered by $\mathcal{S}'$ (i.e., there is some $i \in \mathcal{S}'$ such that $d(i, j) \leq 3R$) but becomes unconnected due to the removal of i_1 or i_2. We say that j is a bad vertex iff $\mathrm{E}[X_j] \geq \gamma$. Otherwise, vertex j is said to be good. Note that $\sum_{j \in V} X_j \leq 2\epsilon n$ with probability one. Thus, there can be at most $2\epsilon n/\gamma$ bad vertices. Let T be the set of all good vertices. Then

$$|T| \geq n - 2\epsilon n/\gamma = (1 - \gamma)n.$$

Algorithm 8 RMatCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ .
2: Let $\mathcal{P}' := \left\{z \in [0,1]^V : z(U) \leq r_{\mathcal{M}}(U)\ \forall U \subseteq V\ \wedge\ z(F_i) \leq 1\ \ \forall i \in V'\right\}$
3: Find a basic solution $Y \in \mathcal{P}'$ which maximizes the linear function $f : [0,1]^V \to \mathbb{R}$ defined as
       $f(z) := \sum_{j \in V'} c_j \sum_{i \in F_j} z_i$  for $z \in [0,1]^V$.
4: return $\mathcal{S} = \{i \in V : Y_i = 1\}$.
```

By Proposition 14, $\Pr[j \text{ is covered by } \mathcal{S}'] \geq p_j$. For any $j \in T$, we have

$$\begin{aligned}\Pr[j \in \mathcal{C}] &= \Pr[j \text{ is covered by } \mathcal{S}' \wedge X_j = 0]\\ &= \Pr[j \text{ is covered by } \mathcal{S}'] - \Pr[j \text{ is covered by } \mathcal{S}' \wedge X_j = 1]\\ &\geq \Pr[j \text{ is covered by } \mathcal{S}'] - \Pr[X_j = 1]\\ &\geq p_j - \gamma.\end{aligned}$$

This concludes the first part of Theorem 4 for the FRKnapCenter problem.

5 The Matroid Center problems with outliers

In this section, we will first give a tight 3-approximation algorithm for the RMatCenter problem, improving upon the 7-approximation algorithm by Chen et. al. [3]. Then we study the FRMatCenter problem and give a proof for the second part of Theorem 4.

5.1 The robust matroid center problem

Suppose $\mathcal{I} = (V, d, \mathcal{M}, t)$ is an instance the RMatCenter problem with the optimal radius R. Let $r_{\mathcal{M}}$ denote the rank function of $\mathcal{M}$. Consider the polytope $\mathcal{P}_{\mathsf{RMatCenter}}$ containing points (x, y) satisfying constraints (1)–(4), and the matroid rank constraints:

$$y(U) \leq r_{\mathcal{M}}(U), \quad \forall U \subseteq V. \tag{8}$$

Since R is the optimal radius, it is not difficult to check that $\mathcal{P}_{\mathsf{RMatCenter}} \neq \emptyset$. Let us pick any fractional solution $(x, y) \in \mathcal{P}_{\mathsf{RMatCenter}}$. The next step is to round (x, y) into an integral solution. Our 3-approximation algorithm is summarized in Algorithm 8.

Analysis. Again, by construction, the clusters F_i are pairwise disjoint for $i \in V'$. Note that $\mathcal{P}'$ is the matroid intersection polytope between $\mathcal{M}$ and another partition matroid polytope saying that at most one item per set F_i for $i \in V'$ can be chosen. Moreover, $y \in \mathcal{P}'$ implies that $\mathcal{P}' \neq \emptyset$. Thus, $\mathcal{P}'$ has integral extreme points and optimizing over $\mathcal{P}'$ can be done in polynomial time. Note that the solution $\mathcal{S}$ is feasible as it satisfies the matroid constraint. The correctness of RMatCenterRound follows immediately by the following two propositions.

▶ **Proposition 17.** *There are at least $f(Y)$ vertices in V that are at distance at most $3R$ from some open center in $\mathcal{S}$.*

▶ **Proposition 18.** *We have that $f(Y) \geq t$.*

This analysis proves the second part of Theorem 1.

Algorithm 9 RoundSinglePoint $(y, \vec{r})$

```
$\delta^* \leftarrow \max\{\delta : z \in \mathcal{P}_{\mathcal{M}}; z_v = y_v + \delta r_v \ \forall v \in V\}$
$y' \leftarrow y + \delta^* \vec{r}$
return $(y', \delta^*)$
```

5.2 The fair robust matroid center problem

In this section, we consider the FRMatCenter problem. It is not difficult to modify and randomize algorithm RMCenterRound so that it would return a random solution satisfying both the fairness guarantee and matroid constraint, and preserving the coverage constraint *in expectation*. This can be done by randomly picking Y inside $\mathcal{P}'$. However, if we want to obtain some concrete guarantee on the coverage constraint, we may have to (slightly) violate either the matroid constraint or the fairness guarantee. We leave it as an open question whether there exists a true approximation algorithm for this problem.

We will start with a pseudo-approximation algorithm which always returns a basis of $\mathcal{M}$ plus at most one extra center. Our algorithm is quite involved. We first carefully round a fractional solution inside a matroid intersection polytope into a (random) point with a special property: the unrounded variables form a single path connecting some clusters and tight matroid rank constraints. Next, rounding this point will ensure that all but one cluster have an open center. Then opening one extra center is sufficient to cover at least t clients.

Finally, using a similar preprocessing step similar to the one in Section 4.2.3, we can correct the solution by removing the extra center without affecting the fairness and coverage guarantees by too much. This algorithm concludes Theorem 4.

5.2.1 A pseudo-approximation algorithm

Suppose $\mathcal{I} = (V, d, \mathcal{M}, t, \vec{p})$ is an instance the robust matroid center problem with the optimal radius R. Let $r_{\mathcal{M}}$ denote the rank function of $\mathcal{M}$ and $\mathcal{P}_{\mathcal{M}}$ be the matroid base polytope of $\mathcal{M}$. Consider the polytope $\mathcal{P}_{\mathsf{FRMatCenter}}$ containing points (x, y) satisfying constraints (1)–(4), the fairness constraint (5), and the matroid constraints (8). Using similar arguments as in the proof of Proposition 9, we can show that $\mathcal{P}_{\mathsf{FRMatCenter}}$ is a valid relaxation.

▶ **Proposition 19.** *We have that $\mathcal{P}_{\mathsf{FRMatCenter}} \neq \emptyset$.*

Our algorithm will use the following rounding operation iteratively.

Given a point $y \in \mathcal{P}_{\mathcal{M}}$ and a vector $\vec{r}$, the procedure RoundSinglePoint will move y along direction $\vec{r}$ to a new point $y + \delta^* \vec{r}$ for some maximal $\delta^* > 0$ such that this point still lies in $\mathcal{P}_{\mathcal{M}}$. Note that one can find such a maximal δ^* in polynomial time. We will choose the initial point (x, y) as a vertex of $\mathcal{P}_{\mathsf{FRMatCenter}}$. By Cramer's rule, the entries of y will be rational with both numerators and denominators bounded by $O(2^n)$. The direction vector $\vec{r}$ also has this property by construction. Thus, it is not hard to verify that the maximal value of δ^* for which $y + \delta^* \vec{r} \in \mathcal{P}_{\mathcal{M}}$ is also rational and has both numerator and denominator at most $O(2^n)$ in every iteration. So we can compute δ^* exactly by a simple binary search.

See the appendix for more details.

5.2.2 Analysis of PseudoFRMCenterRound

▶ **Proposition 20.** *In all but the last iteration, the while-loop (lines 4 to 8) of* PseudoFRMCenterRound *preserves the following invariant: if y' lies in the face D of $\mathcal{P}_{\mathcal{M}}$ (w.r.t. all tight matroid rank constraints) at the beginning of the iteration, then $y' \in D$ at the end of this iteration.*

▶ **Proposition 21.** PSEUDOFRMCENTERROUND *terminates in polynomial time.*

▶ **Proposition 22.** *In all iterations, the while-loop (lines* 4 *to* 8*) of* PSEUDOFRMCENTERROUND *satisfies the invariant that* $y'(F_j) \le 1$ *for all* $F_j \in \mathcal{F}$*.*

▶ **Proposition 23.** PSEUDOFRMCENTERROUND *returns a solution* $\mathcal{S}$ *which is some independent set of* $\mathcal{M}$ *plus (at most) one extra vertex in* V*.*

Recall that $\mathcal{C}$ is the (random) set of all clients within radius $3R$ from some center in $\mathcal{S}$, where R is the optimal radius. The following two propositions will conclude our analysis.

▶ **Proposition 24.** $|\mathcal{C}| \ge t$ *with probability one.*

▶ **Proposition 25.** $\Pr[j \in \mathcal{C}] \ge p_j$ *for all* $j \in V$*.*

So far we have proved the following theorem.

▶ **Theorem 26.** PSEUDOFRMCENTERROUND *will return a random solution* $\mathcal{S}$ *such that*

- $\mathcal{S}$ *is the union of some basis of* $\mathcal{M}$ *with (at most) one extra vertex,*
- $|\mathcal{C}| \ge t$ *with probability one,*
- $\Pr[j \in \mathcal{C}] \ge p_j$ *for all* $j \in V$*.*

5.2.3 An algorithm satisfying the matroid constraint exactly

Using a similar technique as in Section 4.2.3, we will develop an approximation algorithm for the FRMatCenter problem which always returns a feasible solution. Let $\epsilon > 0$ a small parameter to be determined. Let $\mathcal{U}$ denote the collection of all possible sets of verticies with size at most $\lceil 1/\epsilon \rceil$ such that U is an independent set of $\mathcal{M}$. Again, we have that $|\mathcal{U}| \le n^{O(1/\epsilon)}$. Suppose R is the optimal radius to our instance. For any $i \in V$, recall that $\mathrm{RBall}(i, U, R)$ is the set of red vertices within radius $3R$ from i.

Consider the *configuration* polytope $\mathcal{P}_{\mathsf{config3}}$ containing points (x, y, q) with the following constraints:

$$\begin{cases} \sum_{U \in \mathcal{U}} q_U = 1 & \\ \sum_{i \in V: d(i,j) \le R} x_{ij}^U \le q_U & \forall j \in V, U \in \mathcal{U} \\ \sum_{U \in \mathcal{U}} \sum_{i \in V: d(i,j) \le R} x_{ij}^U \ge p_j & \forall j \in V \\ x_{ij}^U \le y_i^U & \forall i, j \in V, U \in \mathcal{U} \\ \sum_{i \in W} y_i^U \le q_U r_{\mathcal{M}}(W) & \forall U \in \mathcal{U}, W \subseteq V \\ \sum_{j \in V} \sum_{i \in V: d(i,j) \le R} x_{ij}^U \ge q_U t & \\ y_i^U = 1 & \forall U \in \mathcal{U}, i \in U \\ y_i^U = 0 & \forall U \in \mathcal{U}, i \in V \setminus U, |\mathrm{RBall}(i, U, R)| \ge \epsilon n \\ x_{ij}^U, y_i^U, q_U \ge 0 & \forall i, j \in V, U \in \mathcal{U} \end{cases}$$

We first claim that $\mathcal{P}_{\mathsf{config3}}$ is a valid relaxation polytope for the problem.

▶ **Proposition 27.** *We have that* $\mathcal{P}_{\mathsf{config3}} \ne \emptyset$*.*

Next, let us pick any $(x, y, q) \in \mathcal{P}_{\mathsf{config3}}$ and use Algorithm 10 to round it.

Algorithm 10 FRMCenterRound (x, y, q)

```
Randomly pick a set $U \in \mathcal{U}$ with probability $q_U$
Let $x'_{ij} \leftarrow x^U_{ij}/q_U$ and $y'_i \leftarrow \min\{y^U_i/q_U, 1\}$
$\mathcal{S}' \leftarrow$ PseudoFRMCenterRound $(x', y')$
Let $i^*$ be the "extra" vertex in $\mathcal{S}'$.
return $\mathcal{S} = \mathcal{S}' \setminus \{i\}$
```

Analysis. We are now ready to prove the second part of Theorem 4. Let us fix any $\gamma > 0$ and set $\epsilon := \gamma^2$. Also, let $\mathcal{E}(U)$ denote the event that $U \in \mathcal{U}$ is picked in the algorithm. Note that (x', y') satisfies the following inequalities:

$$\sum_{j \in V} \sum_{i \in V : d(i,j) \leq R} x'_{ij} \geq t,$$

$$\sum_{i \in V : d(i,j) \leq R} x'_{ij} \leq 1, \quad \forall j \in V,$$

$$\sum_{i \in V : d(i,j) \leq R} x'_{ij} = \sum_{i \in V : d(i,j) \leq R} x_{ij}/q_U, \quad \forall j \in V,$$

$$x'_{ij} \leq y'_i, \quad \forall i, j \in V,$$

$$\sum_{i \in W} y'_i \leq r_{\mathcal{M}}(W), \quad \forall W \subseteq V.$$

Moreover, $y'_i = 1$ for all $i \in U$ and $y'_i = 0$ for all $i \in V \setminus U$ and $\text{RBall}(i, U, R) \geq \epsilon n$.

Recall that the algorithm PseudoFRMCenterRound will return a solution $\mathcal{S}'$ is the union of a basis of $\mathcal{M}$ with an extra center i^*. Moreover, we have $0 < y'_{i^*} < 0$, which implies that $i^* \notin U$. Thus, by removing i^* from $\mathcal{S}'$, we ensure that the resulting set is a basis of $\mathcal{M}$ with probability one.

Now we shall prove the coverage guarantee. By Theorem 26, $\mathcal{S}'$ covers at least t vertices within radius $3R$. If a vertex is blue, it can always be connected to some center in U; and hence, it is not affected by the removal of i_1, i_2. Because each of i^* can cover at most ϵn other red vertices, we have

$$|\mathcal{C}| \geq t - \epsilon n = 1 - \gamma^2 n.$$

For any $j \in V$, let X_j be the random indicator for the event that j is covered by $\mathcal{S}'$ (i.e., there is some $i \in \mathcal{S}'$ such that $d(i, j) \leq 3R$) but becomes unconnected due to the removal of i^*. We say that j is a bad vertex iff $\mathrm{E}[X_j] \geq \gamma$. Otherwise, vertex j is said to be good. Again, $\sum_{j \in V} X_j \leq \epsilon n$ with probability one. Thus, there can be at most $\epsilon n/\gamma$ bad vertices. Let T be the set of all good vertices. Then

$$|T| \geq n - \epsilon n/\gamma = (1 - \gamma)n.$$

By Theorem 26, $\Pr[j \text{ is covered by } \mathcal{S}'] \geq p_j$. So, for any $j \in T$, we have

$$\Pr[j \in \mathcal{C}] \geq \Pr[j \text{ is covered by } \mathcal{S}'] - \Pr[X_j = 1] \geq p_j - \gamma.$$

Acknowledgements. We thank the APPROX 2017 referees for their helpful suggestions.

References

1 Deeparnab Chakrabarty, Prachi Goyal, and Ravishankar Krishnaswamy. The non-uniform k-center problem. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*, volume 55, pages 67:1–67:15, 2016.

2 Moses Charikar, Samir Khuller, David M. Mount, and Giri Narasimhan. Algorithms for facility location problems with outliers. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'01, pages 642–651, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics. URL: `http://dl.acm.org/citation.cfm?id=365411.365555`.

3 Danny Z. Chen, Jian Li, Hongyu Liang, and Haitao Wang. Matroid and knapsack center problems. *Integer Programming and Combinatorial Optimization: 16th International Conference, IPCO 2013, Valparaíso, Chile, March 18-20, 2013. Proceedings*, pages 110–122, 2013. `doi:10.1007/978-3-642-36694-9_10`.

4 David Harris, Thomas Pensyl, Aravind Srinivasan, and Khoa Trinh. Fairness in resource allocation and slowed-down dependent rounding. Manuscript, 2017.

5 Dorit S. Hochbaum and David B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *J. ACM*, 33(3):533–550, May 1986. `doi:10.1145/5925.5933`.

6 Wen-Lian Hsu and George L. Nemhauser. Easy and hard bottleneck location problems. *Discrete Applied Mathematics*, 1(3):209–215, 1979. `doi:10.1016/0166-218X(79)90044-1`.

7 Ravishankar Krishnaswamy, Amit Kumar, Viswanath Nagarajan, Yogish Sabharwal, and Barna Saha. The matroid median problem. In *Proceedings of the annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1117–1130. SIAM, 2011.

8 Lap-Chi Lau, R. Ravi, and Mohit Singh. *Iterative Methods in Combinatorial Optimization*. Cambridge University Press, New York, NY, USA, 1st edition, 2011.

9 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.

10 Chaitanya Swamy. Improved approximation algorithms for matroid and knapsack median problems and applications. In *APPROX/RANDOM 2014*, volume 28, pages 403–418, 2014.

A Details of the pseudo-approximation for FRMatCenter

The main algorithm is summarized in Algorithm 11, which can round any *vertex* point $(x, y) \in \mathcal{P}_{\mathsf{FRMatCenter}}$. Basically, we will round y iteratively. In each round, we construct a (multi)-bipartite graph where vertices on the left side are the disjoint sets $O_1, O_2, \ldots$ in Corollary 8. Vertices on the right side are corresponding to the disjoint sets $F_1, F_2, \ldots$ returned by RFiltering. Now each edge of the bipartite graph, connecting O_i and F_j, represents some unrounded variable $y_v \in (0, 1)$ where $v \in O_i$ and $v \in F_j$. See Figure 1.

Then we carefully pick a cycle (path) on this graph and round variables on the edges of this cycle (path). This is done by subroutines RoundCycle, RoundSinglePath, and RoundTwoPaths. See Figures 2, 3, and 4. Basically, these procedures will first choose a direction $\vec{r}$ which alternatively increases and decreases the variables on the cycle (path) so that (i) all tight matroid constraints are preserved and (ii) the number of (fractionally) covered clients is also preserved. Now we randomly move y along $\vec{r}$ or $-\vec{r}$ using procedure RoundSinglePoint to ensure that all the marginal probabilities are preserved.

Finally, all the remaining, fractional variables will form one path on the bipartite graph. We round these variables by the procedure RoundFinalPath which exploits the integrality of any face of a matroid intersection polytope. Then, to cover at least t clients, we may need to open one extra facility.

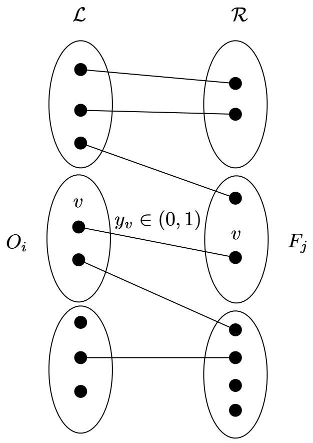


Figure 1 Construction of the multi-bipartite graph $H = (\mathcal{L}, \mathcal{R}, E_H)$ in the main algorithm.

Algorithm 11 PseudoFRMCenterRound (x, y)

```
1: $(V', \vec{c}) \leftarrow$ RFiltering $(x, y)$ and let $\mathcal{F} \leftarrow \{F_j : j \in V'\}$
2: Set $y'_i \leftarrow x_{ij}$ for all $j \in V', i \in F_j$
3: Set $y'_i \leftarrow 0$ for all $i \in V \setminus \bigcup_{j \in V'} F_j$
4: while $y'$ still contains some fractional values do
5:     Note that $y' \in \mathcal{P}_{\mathcal{M}}$. Compute the disjoint sets $O_1, \ldots, O_t$ and constants $b_{O_1}, \ldots, b_{O_t}$
       as in Corollary 8.
6:     Let $O_0 \leftarrow V \setminus \bigcup_{i=1}^{t} O_i$ and $F_0 \leftarrow V \setminus \bigcup_{j \in V'} F_j$
7:     Construct a multi-bipartite graph $H = (\mathcal{L}, \mathcal{R}, E_H)$ where
         - each vertex $i \in \mathcal{L}$, where $\mathcal{L} = \{0, \ldots, t\}$, is corresponding to the set $O_i$
         - each vertex $j \in \mathcal{R}$, where $\mathcal{R} = \{0\} \cup \{k : F_k \in \mathcal{F}\}$, is corresponding to the set $F_j$
         - for each vertex $v \in V$ such that $y_v \in (0,1)$: if $v$ belongs to some set $O_i$ and $F_j$,
           add an edge $e$ with label $v$ connecting $i \in \mathcal{L}$ and $j \in \mathcal{R}$.
8:     Check the following cases (in order):
         - Case 1: $H$ contains a cycle. Let $\vec{v} = (v_1, v_2, \ldots, v_{2\ell})$ be the sequence of edge labels
           on this cycle. Update $y' \leftarrow$ RoundCycle$(y', \vec{v})$ and go to line 4.
         - Case 2: $H$ contains a maximal path with one endpoint in $\mathcal{L}$ and the other in $\mathcal{R}$.
           Let $\vec{v} = (v_1, v_2, \ldots, v_{2\ell+1})$ be the sequence of edge labels on this path. Update
           $y' \leftarrow$ RoundSinglePath$(y', \vec{v})$ and go to line 4.
         - Case 3: There are at least 2 distinct maximal paths (not necessarily disjoint) having
           both endpoints in $\mathcal{R}$. Let $\vec{v}_1, \vec{v}_2$ be the sequences of edge labels on these two paths.
           Update $y' \leftarrow$ RoundTwoPaths$(y', \vec{v}_1, \vec{v}_2, \vec{c})$ and go to line 4.
         - The remaining case: all edges in $H$ form a single path with both endpoints
           in $\mathcal{R}$. Let $(v_1, v_2, \ldots, v_{2\ell})$ be the sequence of edge labels on this path. Let
           $Y \leftarrow$ RoundFinalPath$(y', \vec{v})$ and exit the loop.
9: return $\mathcal{S} = \{i \in V : Y_i = 1\}$.
```

Algorithm 12 RoundCycle $(y', \vec{v})$

```
1: Initialize $\vec{r} = \vec{0}$, then set $r_{v_j} = (-1)^j$ for $j = 1, 2, \ldots, |\vec{v}|$
2: $(y_1, \delta_1) \leftarrow$RoundSinglePoint$(y', \vec{r})$
3: return $y_1$
```

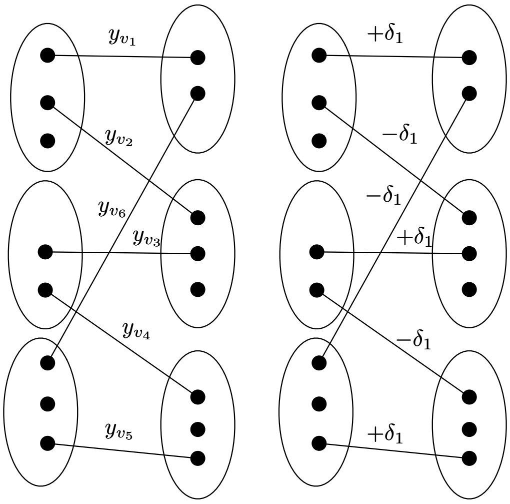


Figure 2 The left part shows a cycle. The right part shows how the variables on the cycle are being changed by RoundCycle.

Algorithm 13 RoundSinglePath $(y', \vec{v})$

```
Initialize $\vec{r} = \vec{0}$, then set $r_{v_j} = (-1)^{j+1}$ for $j = 1, 2, \ldots, |\vec{v}|$
$(y_1, \delta_1) \leftarrow$RoundSinglePoint$(y', \vec{r})$
**return** $y_1$
```

Algorithm 14 RoundTwoPaths $(y', \vec{v}, \vec{v}', \vec{c})$

```
WLOG, suppose $j_1, j_2 \in \mathcal{R}$ are endpoints of $v_1, v_{2\ell}$ of the path $\vec{v}$ respectively and $c_{j_1} \geq c_{j_2}$
WLOG, suppose $j'_1, j'_2 \in \mathcal{R}$ are endpoints of $v'_1, v'_{2\ell'}$ of the path $\vec{v}'$ respectively and $c_{j'_1} \geq c_{j'_2}$
$\Delta_1 \leftarrow c_{j_1} - c_{j_2}$;   $\Delta_2 \leftarrow c_{j'_1} - c_{j'_2}$;   $\vec{r} \leftarrow \vec{0}$
$V_1^+ \leftarrow \{v_1, v_3, \ldots, v_{2\ell-1}\}; V_1^- \leftarrow \{v_2, v_4, \ldots, v_{2\ell}\}$
$V_2^+ \leftarrow \{v'_2, v'_4, \ldots, v'_{2\ell'}\}; V_2^- \leftarrow \{v'_1, v'_3, \ldots, v'_{2\ell'-1}\}$
**for each** $v \in V_1^+$: $r_v \leftarrow r_v + 1$; **for each** $v \in V_1^-$: $r_v \leftarrow r_v - 1$
**for each** $v \in V_2^+$: $r_v \leftarrow r_v + \Delta_1/\Delta_2$; **for each** $v \in V_2^-$: $r_v \leftarrow r_v - \Delta_1/\Delta_2$
$(y_1, \delta_1) \leftarrow$RoundSinglePoint$(y', \vec{r})$
$(y_2, \delta_2) \leftarrow$RoundSinglePoint$(y', -\vec{r})$
With probability $\delta_1/(\delta_1 + \delta_2)$: **return** $y_2$
With remaining probability $\delta_2/(\delta_1 + \delta_2)$: **return** $y_1$
```

Algorithm 15 RoundFinalPath $(y, \vec{v})$

```
$\mathcal{P}_1 \leftarrow \left\{z \in [0,1]^V : z(U) \leq r_{\mathcal{M}}(U)\ \forall U \subseteq V \wedge z(O_i) = b_{O_i}\ \forall i \in \mathcal{L} \setminus \{0\} \wedge z_i = 0\ \forall i : y_i = 0\right\}$
$\mathcal{P}_2 \leftarrow \{z \in [0,1]^V : z(F_j) = y(F_j)\ \forall j \in V' \setminus J \wedge z(F_j) \leq 1\ \forall j \in J\}$, where $J \subseteq \mathcal{R}$ is the set of vertices in $\mathcal{R}$ on the path $\vec{v}$.
Pick an arbitrary extreme point $\hat{y}$ of $\mathcal{P}' = \mathcal{P}_1 \cap \mathcal{P}_2$
**for each** $j \in \mathcal{R}$ and $j$ is on the path $\vec{v}$: if $\hat{y}(F_j) = 0$, pick an arbitrary $u \in F_j$ and set $\hat{y}_u \leftarrow 1$.
**return** $\hat{y}$
```

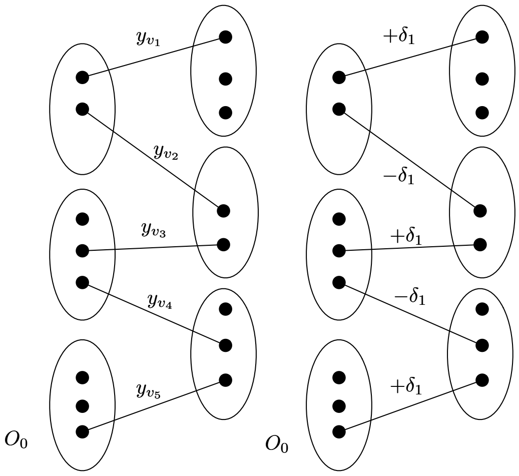


Figure 3 The left part shows a single path. The right part shows how the variables on the path are being changed by ROUNDSINGLEPATH.

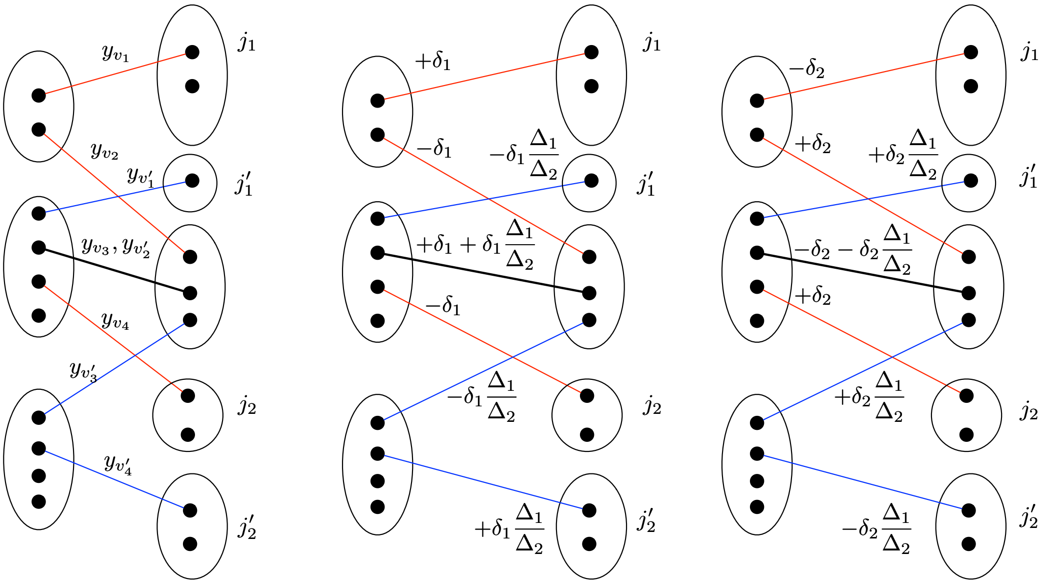

y_{v_1}
j_1
y_{v_2}
$y_{v'_1}$
j'_1
$y_{v_3}, y_{v'_2}$
y_{v_4}
$y_{v'_3}$
j_2
$y_{v'_4}$
j'_2
$+\delta_1$
$-\delta_1$
$-\delta_1 \frac{\Delta_1}{\Delta_2}$
$+\delta_1 + \delta_1 \frac{\Delta_1}{\Delta_2}$
$-\delta_1$
$-\delta_1 \frac{\Delta_1}{\Delta_2}$
$+\delta_1 \frac{\Delta_1}{\Delta_2}$
$-\delta_2$
$+\delta_2$
$+\delta_2 \frac{\Delta_1}{\Delta_2}$
$-\delta_2 - \delta_2 \frac{\Delta_1}{\Delta_2}$
$+\delta_2$
$+\delta_2 \frac{\Delta_1}{\Delta_2}$
$-\delta_2 \frac{\Delta_1}{\Delta_2}$

Figure 4 The left part shows an example of two distinct maximal paths chosen in Case 3. The black edge is common in both paths. The middle and right parts are two possibilities of rounding y. With probability $\delta_1/(\delta_1 + \delta_2)$, the strategy in the right part is adopted. Otherwise, the strategy in the middle part is chosen.

Streaming Algorithms for Maximizing Monotone Submodular Functions under a Knapsack Constraint*

Chien-Chung Huang[1], Naonori Kakimura[†2], and Yuichi Yoshida[‡3]

1 **CNRS, École Normale Supérieure, Paris, France**
`villars@gmail.com`

2 **Department of Mathematics, Keio University, Yokohama, Japan**
`kakimura@math.keio.ac.jp`

2 **National Institute of Informatics, Tokyo, Japan**
`yyoshida@nii.ac.jp`

Abstract

In this paper, we consider the problem of maximizing a monotone submodular function subject to a knapsack constraint in the streaming setting. In particular, the elements arrive sequentially and at any point of time, the algorithm has access only to a small fraction of the data stored in primary memory. For this problem, we propose a $(0.363 - \varepsilon)$-approximation algorithm, requiring only a single pass through the data; moreover, we propose a $(0.4 - \varepsilon)$-approximation algorithm requiring a constant number of passes through the data. The required memory space of both algorithms depends only on the size of the knapsack capacity and ε.

1998 ACM Subject Classification G.2.1 Combinatorics

Keywords and phrases submodular functions, single-pass streaming, multiple-pass streaming, constant approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.11

1 Introduction

A set function $f : 2^E \to \mathbb{R}_+$ on a ground set E is called *submodular* if it satisfies the *diminishing marginal return property*, i.e., for any subsets $S \subseteq T \subsetneq E$ and $e \in E \setminus T$, we have

$$f(S \cup \{e\}) - f(S) \geq f(T \cup \{e\}) - f(T).$$

A function is *monotone* if $f(S) \leq f(T)$ for any $S \subseteq T$. Submodular functions play a fundamental role in combinatorial optimization, as they capture rank functions of matroids, edge cuts of graphs, and set coverage, just to name a few examples. Besides their theoretical interests, submodular functions have attracted much attention from the machine learning community because they can model various practical problems such as online advertising [1, 11, 18], sensor location [12], text summarization [16, 17], and maximum entropy sampling [14].

[*] A full version of the paper is available at `http://www.di.ens.fr/~cchuang/work/streaming_knapsack.pdf`.

[†] Partly supported by JST ERATO Grant Number JPMJER1305, and JSPS KAKENHI Grant Number JP17K00028.

[‡] Partly supported by JST ERATO Grant Number JPMJER1305 and JSPS KAKENHI Grant Number JP17H04676

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 11; pp. 11:1–11:14
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Many of the aforementioned applications can be formulated as the maximization of a monotone submodular function under a knapsack constraint. In this problem, we are given a monotone submodular function $f : 2^E \to \mathbb{R}_+$, a size function $c : E \to \mathbb{N}$, and an integer $K \in \mathbb{N}$, where $\mathbb{N}$ denotes the set of positive integers. The problem is defined as

$$\text{maximize} \quad f(S) \quad \text{subject to} \quad c(S) \leq K, \tag{1}$$

where we denote $c(S) = \sum_{e \in S} c(e)$ for a subset $S \subseteq E$. Throughout this paper, we assume that every item $e \in E$ satisfies $c(e) \leq K$ as otherwise we can simply discard it. Note that, when $c(e) = 1$ for every item $e \in E$, the constraint coincides with a cardinality constraint.

The problem of maximizing a monotone submodular function under a knapsack constraint is classical and well-studied. First introduced by Wolsey [20], the problem is known to be NP-hard but can be approximated within the factor of (close to) $1 - 1/e$; see e.g., [3, 10, 13, 8, 19].

In some applications, the amount of input data is much larger than the main memory capacity of individual computers. In such a case, we need to process data in a *streaming* fashion. That is, we consider the situation where each item in the ground set E arrives sequentially, and we are allowed to keep only a small number of the items in memory at any point. This setting effectively rules out most of the techniques in the literature, as they typically require random access to the data. In this work, we also assume that the function oracle of f is available at any point of the process. Such an assumption is standard in the submodular function literature and in the context of streaming setting [2, 7, 21]. Badanidiyuru *et al.* [2] discuss several interesting and useful functions where the oracle can be implemented using a small subset of the entire ground set E.

We note that the problem, under the streaming model, has so far not received its deserved attention in the community. Prior to the present work, we are aware of only two: for the special case of cardinality constraint, Badanidiyuru *et al.* [2] gave a single-pass $(1/2 - \varepsilon)$-approximation algorithm; for the general case of a knapsack constraint, Yu *et al.* [21] gave a single-pass $(1/3 - \varepsilon)$-approximation algorithm, both using $O(K \log(K)/\varepsilon)$ space.

We now state our contribution.

▶ **Theorem 1.** *For the problem* (1)*,*

1. *there is a single-pass streaming algorithm with approximation ratio* $4/11 - \varepsilon \approx 0.363 - \varepsilon$*,*
2. *there is a multiple-pass streaming algorithm with approximation ratio* $2/5 - \varepsilon = 0.4 - \varepsilon$*.*

Both algorithms use $O(K \cdot \mathrm{poly}(\varepsilon^{-1})\mathrm{polylog}(K))$ *space.*

Our Technique

We begin by a straightforward generalization of the algorithm of [2] for the special case of cardinality constraint (Section 2). This algorithm proceeds by adding a new item into the current set only if its marginal-ratio (its marginal return with respect to the current set divided by its size) exceeds a certain threshold. This algorithm performs well when all items in OPT are relatively small in size, where OPT is an optimal solution. However, in general, it only gives $(1/3 - \varepsilon)$-approximation. Note that this technique can be regarded as a variation of the one in [21]. To obtain better approximation ratio, we need new ideas.

The difficulty in improving this algorithm lies in the following case: A new arriving item that is relatively large in size, passes the marginal-ratio threshold, and is part of OPT, but its addition would cause the current set to exceed the capacity K. In this case, we are forced to throw it away, but in doing so, we are unable to bound the ratio of the function value of the current set against that of OPT properly.

We propose a branching procedure to overcome this issue. Roughly speaking, when the function value of the current set is large enough (depending on the parameters), we create a secondary set. We add an item to the secondary set only if it passes the marginal-ratio threshold (with respect to the original set) but its addition to the original set would violate the size constraint. In the end, whichever set achieves the higher value is returned. In a way, the secondary set serves as a "back-up" with enough space in case the original set does not have it, and this allows us to bound the ratio properly. Sections 3 and 4 are devoted to explaining this branching algorithm, which gives $(4/11 - \varepsilon)$-approximation with a single pass.

We note that the main bottleneck of the above singe-pass algorithm lies in the situation where there is a large item in OPT whose size exceeds $K/2$. In Section 5, we show that we can first focus on only the large items (more specifically, those items whose size differ from the largest item in OPT by $(1+\varepsilon)$ factor) and choose $O(1)$ of them so that at least one of them, along with the rest of OPT (excluding the largest item in it), gives a good approximation to $f(\mathrm{OPT})$. Then in the next pass, we can apply a modified version of the original single-pass algorithm to collect small items. This multiple-pass algorithm gives a $(2/5 - \varepsilon)$-approximation.

We remark that the proofs of some lemmas and theorems are omitted due to the page limitation, which can be found in the full version of this paper.

Related Work

Maximizing a monotone submodular function subject to various constraints is a subject that has been extensively studied in the literature. We are unable to give a complete survey here and only highlight the most representative and relevant results. Besides a knapsack constraint or a cardinality constraint mentioned above, the problem has also been studied under (multiple) matroid constraint(s), p-system constraint, multiple knapsack constraints. See [4, 9, 13, 8, 15] and the references therein. In the streaming setting, other than the knapsack constraint that we have discussed before, there are also works considering a matroid constraint. Chakrabarti and Kale [5] gave 1/4-approximation; Chekuri *et al.* [7] gave the same ratio. Very recently, for the special case of partition matroid, Chan *et al.* [6] improved the ratio to 0.3178.

Notation

For a subset $S \subseteq E$ and an element $e \in E$, we use the shorthand $S + e$ and $S - e$ to stand for $S \cup \{e\}$ and $S \setminus \{e\}$, respectively. For a function $f : 2^E \to \mathbb{R}$, we also use the shorthand $f(e)$ to stand for $f(\{e\})$. The *marginal return* of adding $e \in E$ with respect to $S \subseteq E$ is defined as $f(e \mid S) = f(S+e) - f(S)$. We frequently use the following, which is immediate from the diminishing marginal return property:

▶ **Proposition 2.** *Let $f : 2^E \to \mathbb{R}_+$ be a monotone submodular function. For two subsets $S \subseteq T \subseteq E$, it holds that $f(T) \leq f(S) + \sum_{e \in T \setminus S} f(e \mid S)$.*

2 Single-Pass $(1/3 - \varepsilon)$-Approximation Algorithm

In this section, we present a simple $(1/3 - \varepsilon)$-approximation algorithm that generalizes the algorithm for a cardinality constraint in [2]. This algorithm will be incorporated into several other algorithms introduced later.

Algorithm 1

```
procedure MarginalRatioThresholding($\alpha, v$)                ▷ $\alpha \in (0,1], v \in \mathbb{R}_+$
    $S := \emptyset$.
    while item $e$ is arriving do
        if $\frac{f(e|S)}{c(e)} \geq \frac{\alpha v - f(S)}{K - c(S)}$ and $c(S+e) \leq K$ then $S := S + e$.
    return $S$.
```

2.1 Thresholding Algorithm with Approximate Optimal Value

In this subsection, we present an algorithm MarginalRatioThresholding, which achieves (almost) 1/3-approximation given a (good) approximation v to $f(\mathrm{OPT})$ for an optimal solution OPT. This assumption is removed in Section 2.2.

Given a parameter $\alpha \in (0,1]$ and $v \in \mathbb{R}_+$, MarginalRatioThresholding attempts to add a new item $e \in E$ to the current set $S \subseteq E$ if its addition does not violate the knapsack constraint and e passes the *marginal-ratio threshold condition*, i.e.,

$$\frac{f(e \mid S)}{c(e)} \geq \frac{\alpha v - f(S)}{K - c(S)}. \tag{2}$$

The detailed description of MarginalRatioThresholding is given in Algorithm 1.

Throughout this subsection, we fix $\tilde{S} = \mathsf{MarginalRatioThresholding}(\alpha, v)$ as the output of the algorithm. Then, we have the following lemma.

▶ **Lemma 3.** *The following hold:*

(1) *During the execution of the algorithm, the current set $S \subseteq E$ always satisfies $f(S) \geq \alpha v c(S)/K$. Moreover, if an item $e \in E$ passes the condition* (2) *with the current set S, then $f(S+e) \geq \alpha v c(S+e)/K$.*

(2) *If an item $e \in E$ fails the condition* (2)*, i.e., $\frac{f(e|S)}{c(e)} < \frac{\alpha v - f(S)}{K - c(S)}$, then we have $f(e \mid \tilde{S}) < \alpha v c(e)/K$.*

An item $e \in \mathrm{OPT}$ is not added to $\tilde{S}$ if either e does not pass the condition (2), or its addition would cause the size of S to exceed the capacity K. We name the latter condition as follows:

▶ **Definition 4.** An item $e \in \mathrm{OPT}$ is called *bad* if e passes the condition (2) but the total size exceeds K when added, i.e., $f(e \mid S) \geq \frac{\alpha v - f(S)}{K - c(S)}$, $c(S+e) > K$ and $c(S) \leq K$, where S is the set we have just before e arrives.

The following lemma says that, if there is no bad item, then we obtain a good approximation.

▶ **Lemma 5.** *If $v \leq f(\mathrm{OPT})$ and there have been no bad item, then $f(\tilde{S}) \geq (1-\alpha)v$ holds.*

Proof. By the submodularity and the monotonicity, we have $v \leq f(\mathrm{OPT}) \leq f(\mathrm{OPT} \cup \tilde{S}) \leq f(\tilde{S}) + \sum_{e \in \mathrm{OPT} \setminus \tilde{S}} f(e \mid \tilde{S})$. Since we have no bad item, $f(e \mid \tilde{S}) \leq \alpha v c(e)/K$ for any $e \in \mathrm{OPT} \setminus \tilde{S}$ by Lemma 3 (2). Hence, we have $v \leq f(\tilde{S}) + \alpha v$, implying $f(\tilde{S}) \geq (1-\alpha)v$. ◀

Consider an algorithm Singleton, which takes the best singleton as shown in Algorithm 2. If some item $e \in \mathrm{OPT}$ is bad, then, together with $\tilde{S}' = \mathsf{Singleton}()$, we can achieve (almost) 1/3-approximation.

▶ **Theorem 6.** *We have $\max\{f(\tilde{S}), f(\tilde{S}')\} \geq \min\{\alpha/2, 1-\alpha\}v$. The right-hand side is maximized to $v/3$ when $\alpha = 2/3$.*

Algorithm 2

```
procedure Singleton()
    $S := \emptyset$.
    while item $e$ is arriving do
        if $f(e) > f(S)$ then $S := \{e\}$.
    return $S$.
```

Algorithm 3

```
procedure DynamicMRT($\varepsilon, \alpha$)                          ▷ $\varepsilon, \alpha \in (0, 1]$
    $\mathcal{V} := \{(1+\varepsilon)^i \mid i \in \mathbb{Z}_+\}$.
    For each $v \in \mathcal{V}$, set $S_v := \emptyset$.
    while item $e$ is arriving do
        $m := \max\{m, f(e)\}$.
        $\mathcal{I} := \{v \in \mathcal{V} \mid m \leq v \leq Km/\alpha\}$.
        Delete $S_v$ for each $v \notin \mathcal{I}$.
        for each $v \in \mathcal{I}$ do
            if $\frac{f(e|S_v)}{c(e)} \geq \frac{\alpha v - f(S_v)}{K - c(S_v)}$ and $c(S_v + e) \leq K$ then $S_v := S_v + e$.
    return $S_v$ for $v \in \mathcal{I}$ that maximizes $f(S_v)$.
```

Proof. If there exists no bad item, we have $f(\tilde{S}) \geq (1-\alpha)v$ by Lemma 5. Suppose that we have a bad item $e \in E$. Let $S_e \subseteq E$ be the set just before e arrives in MarginalRatioThresholding. Then, we have $f(S_e + e) \geq \alpha v c(S_e + e)/K$ by Lemma 3 (1). Since $c(S_e + e) > K$, this means $f(S_e + e) \geq \alpha v$. Since $f(S_e + e) \leq f(S_e) + f(e)$ by submodularity, one of $f(S_e)$ and $f(e)$ is at least $\alpha v/2$. Thus $f(\tilde{S}) \geq f(S_e) \geq \alpha v/2$ or $f(e) \geq \alpha v/2$. ◀

Therefore, if we have $v \in \mathbb{R}_+$ with $v \leq f(\mathrm{OPT}) \leq (1+\varepsilon)v$, the algorithm that runs MarginalRatioThresholding$(2/3, v)$ and Singleton() in parallel and chooses the better output has the approximation ratio of $\frac{1}{3(1+\varepsilon)} \geq 1/3 - \varepsilon$. The space complexity of the algorithm is clearly $O(K)$.

2.2 Dynamic Updates

MarginalRatioThresholding requires a good approximation to $f(\mathrm{OPT})$. This requirement can be removed with dynamic updates in a similar way to [2]. We first observe that $\max_{e\in S} f(e) \leq f(\mathrm{OPT}) \leq K \max_{e\in S} f(e)$. So if we are given $m = \max_{e\in S} f(e)$ in advance, a value $v \in \mathbb{R}_+$ with $v \leq f(\mathrm{OPT}) \leq (1+\varepsilon)v$ for $\varepsilon \in (0, 1]$ exists in the guess set $\mathcal{I} = \{(1+\varepsilon)^i \mid m \leq (1+\varepsilon)^i \leq Km, i \in \mathbb{Z}_+\}$. Then, we can run MarginalRatioThresholding for each $v \in \mathcal{I}$ in parallel and choose the best output. As the size of $\mathcal{I}$ is $O(\log(K)/\varepsilon)$, the total space complexity is $O(K \log(K)/\varepsilon)$.

To get rid of the assumption that we are given m in advance, we consider an algorithm, called DynamicMRT, which dynamically updates m to determine the range of guessed optimal values. More specifically, it keeps the (tentative) maximum value $\max f(e)$, where the maximum is taken over the items e arrived so far, and keeps the approximations v in the interval between m and Km/α. The details are provided in Algorithm 3. We have the following guarantee.

▶ **Theorem 7.** *For $\varepsilon \in (0, 1]$, the algorithm that runs* DynamicMRT$(\varepsilon, 2/3)$ *and* Singleton() *in parallel and outputs the better output is a $(1/3-\varepsilon)$-approximation streaming algorithm with a single pass for the problem (1). The space complexity of the algorithm is $O(K \log(K)/\varepsilon)$.*

Algorithm 4

```
1:  procedure BranchingMRT($\varepsilon, \alpha, v, c_1, b$)    ▷ $\varepsilon, \alpha \in (0,1]$, $v \in \mathbb{R}_+$, and $c_1, b \in [0,1/2]$
2:      $S := \emptyset$.
3:      $\lambda := \frac{1}{2}\alpha(1-b)v$.
4:      while item $e$ is arriving do
5:          Delete $e$ with $c(e) > \min\{(1+\varepsilon)c_1, 1/2\}K$.
6:          if $\frac{f(e|S)}{c(e)} \geq \frac{\alpha v - f(S)}{K - c(S)}$ and $c(S+e) \leq K$ then $S := S + e$.
7:          if $f(S) \geq \lambda$ then break // leave the While loop.
8:      Let $\hat{e}$ be the latest added item in $S$.
9:      if $c(S) \geq (1-b)K$ then $S_0' := \{\hat{e}\}$ else $S_0' := S$.
10:     $S' := S_0'$.
11:     while item $e$ is arriving do
12:         Delete $e$ with $c(e) > \min\{(1+\varepsilon)c_1, 1/2\}K$.
13:         if $\frac{f(e|S)}{c(e)} \geq \frac{\alpha v - f(S)}{K - c(S)}$ and $c(S+e) \leq K$ then $S := S + e$.
14:         if $\frac{f(e|S)}{c(e)} \geq \frac{\alpha v - f(S)}{K - c(S)}$ and $c(S+e) > K$ then
15:             if $f(S') < f(S_0' + e)$ then $S' := S_0' + e$.
16:     return $S$ or $S'$ whichever has the larger function value.
```

3 Improved Single-Pass Algorithm for Small-Size Items

Let OPT $= \{o_1, o_2, \ldots, o_\ell\}$ be an optimal solution with $c(o_1) \geq c(o_2) \geq \cdots \geq c(o_\ell)$. The main goal of this section is achieving $(2/5 - \varepsilon)$-approximation, assuming that $c(o_1) \leq K/2$. The case with $c(o_1) > K/2$ will be discussed in Section 4.

3.1 Branching Framework with Approximate Optimal Value

We here provide a framework of a branching algorithm BranchingMRT as Algorithm 4. This will be used with different parameters in Section 3.2.

Let v and c_1 be (good) approximations to $f(\mathrm{OPT})$ and $c(o_1)/K$, respectively, and let $b \leq 1/2$ be a parameter. The value c_1 is supposed to satisfy $c_1 \leq c(o_1)/K \leq (1+\varepsilon)c_1$, and hence we ignore items $e \in E$ with $c(e) > \min\{(1+\varepsilon)c_1, 1/2\}K$. The basic idea of BranchingMRT is to take only items with large marginal ratios, similarly to MarginalRatioThresholding. The difference is that, once $f(S)$ exceeds a threshold λ, where $\lambda = \frac{1}{2}\alpha\,(1-b)\,v$, we store either the current set S or the latest added item as S'. This guarantees that $f(S') \geq \lambda$ and $c(S') \leq (1-b)K$, which means that S' has a large function value and sufficient room to add more elements. We call the process of constructing S' *branching*. We continue to add items with large marginal ratios to the current set S, and if we cannot add an item to S because it exceeds the capacity, we try to add the item to S'. Note that the set S', after branching, can have at most one extra item; but this extra item can be replaced if a better candidate comes along (See line 14–15).

Remark that the sequence of sets S in BranchingMRT is identical to that in MarginalRatioThresholding. Hence, we do not need to run MarginalRatioThresholding in parallel to this algorithm. We say that an item $e \in$ OPT is *bad* if it is bad in the sense of MarginalRatioThresholding, i.e., it satisfies the condition in Definition 4. We have the following two lemmas.

▶ **Lemma 8.** *For a bad item e with $c(e) \leq bK$, let S_e be the set just before e arrives in Algorithm 4. Then $f(S_e) \geq \lambda$ holds. Thus branching has happened before e arrives.*

Proof. Sine e is a bad item, we have $c(S_e) > K - c(e) \geq (1-b)K$. Hence $f(S_e) \geq \alpha(1-b)v \geq \lambda$ by Lemma 3 (1). Since the value of f is non-decreasing during the process, it means that branching has happened before e arrives. ◀

▶ **Lemma 9.** *It holds that* $f(S'_0) \geq \lambda$ *and* $c(S'_0) \leq (1-b)K$.

Proof. We denote by S the set obtained right after leaving the while loop from Line 4. If $c(S) < (1-b)K$, then $f(S'_0) = f(S) \geq \lambda$. Otherwise, since $c(S) \geq (1-b)K$, we have $f(S) \geq \alpha(1-b)v \geq 2\lambda$ by Lemma 3 (1). Hence $f(S'_0) = f(\hat{e}) \geq \lambda$ since $f(S - \hat{e}) < \lambda$ and the submodularity. The second part holds since $c(\hat{e}) \leq K/2 \leq (1-b)K$ by $b \leq 1/2$. ◀

Let $\tilde{S}$ and $\tilde{S}'$ be the final two sets computed by BranchingMRT. Note that we can regard $\tilde{S}$ as the output of MarginalRatioThresholding and $\tilde{S}'$ as the final set obtained by adding at most one item to S'_0.

Observe that the number of bad items depends on the parameter α. As we will show in Section 3.2, by choosing a suitable α, if we have more than two bad items, then the size of $\tilde{S}$ is large enough, implying that $f(\tilde{S})$ is already good for approximation (due to Lemma 3 (1)). Therefore, in the following, we just concentrate on the case when we have at most two bad items.

▶ **Lemma 10.** *Let* α *be a number in* $(0,1]$, *and suppose that we have only one bad item* o_b. *If* $v \leq f(\mathrm{OPT})$ *and* $b \in [c(o_\mathrm{b})/K, (1+\varepsilon)c(o_\mathrm{b})/K]$, *then it holds that*

$$\max\{f(\tilde{S}), f(\tilde{S}')\} \geq \frac{1}{2}\left(1 - \alpha\frac{K - c(o_\mathrm{b})}{2K}\right)v - \frac{\varepsilon\alpha c(o_\mathrm{b})}{4K}v = \left(\frac{1}{2}\left(1 - \alpha\frac{K - c(o_\mathrm{b})}{2K}\right) - O(\varepsilon)\right)v.$$

Proof. Suppose not, that is, suppose that both of $f(\tilde{S})$ and $f(\tilde{S}')$ are smaller than βv, where $\beta = \frac{1}{2}(1 - \alpha\frac{K - c(o_\mathrm{b})}{2K}) - \frac{\alpha c(o_\mathrm{b})}{4K}\varepsilon$. We denote $O_\mathrm{s} = \mathrm{OPT} \setminus \{o_\mathrm{b}\}$.

Since the bad item o_b satisfies $c(o_\mathrm{b}) \leq bK$, it arrives after branching by Lemma 8. By Lemma 9, we have $c(S'_0 + o_\mathrm{b}) \leq K$. Since $f(\tilde{S}')$ is less than βv, we see that $f(S'_0 + o_\mathrm{b}) < \beta v$. Since $f(S'_0) \geq \lambda$,

$$f(\mathrm{OPT}) \leq f(o_\mathrm{b} \mid S'_0) + f(S'_0 \cup O_\mathrm{s}) < (\beta v - \lambda) + f(S'_0 \cup O_\mathrm{s}). \tag{3}$$

Since $S'_0 \subseteq \tilde{S}$, submodularity implies that

$$f(S'_0 \cup O_\mathrm{s}) \leq f(\tilde{S} \cup O_\mathrm{s}) \leq f(\tilde{S}) + \sum_{e \in O_\mathrm{s} \setminus \tilde{S}} f(e \mid \tilde{S}). \tag{4}$$

Since $f(\tilde{S}) < \beta v$ and no item in O_s is bad, (3) and (4) imply by Lemma 3 (2) that

$$\begin{aligned} v \leq f(\mathrm{OPT}) &< (\beta v - \lambda) + f(S'_0 \cup O_\mathrm{s}) < (\beta v - \lambda) + \beta v + \frac{\alpha c(O_\mathrm{s})}{K}v \\ &\leq 2\beta v - \frac{1}{2}\alpha(1-b)v + \alpha\left(1 - \frac{c(o_\mathrm{b})}{K}\right)v. \end{aligned}$$

Therefore, we have

$$\beta > \frac{1}{2}\left(1 + \alpha\frac{2c(o_\mathrm{b})/K - b - 1}{2}\right).$$

Since $b \leq (1+\varepsilon)c(o_\mathrm{b})/K$, we obtain

$$\beta > \frac{1}{2}\left(1 - \frac{(K - c(o_\mathrm{b}))\alpha}{2K}\right) - \frac{\alpha c(o_\mathrm{b})}{4K}\varepsilon,$$

which is a contradiction. This completes the proof. ◀

For the case when we have exactly two bad items, we obtain the following guarantee.

▶ **Lemma 11.** *Let α be a number in $(0,1]$, and suppose that we have exactly two bad items o_{b} and o_{m} with $c(o_{\mathrm{b}}) \geq c(o_{\mathrm{m}})$. If $v \leq f(\mathrm{OPT})$ and $b \in [c(o_{\mathrm{b}})/K, (1+\varepsilon)c(o_{\mathrm{b}})/K]$, then it holds that*

$$\max\{f(\tilde{S}), f(\tilde{S}')\} \geq \frac{1}{3}\left(1 + \alpha\frac{c(o_{\mathrm{m}})}{K}\right)v - \frac{\alpha c(o_{\mathrm{b}})}{3K}\varepsilon v = \left(\frac{1}{3}\left(1 + \alpha\frac{c(o_{\mathrm{m}})}{K}\right) - O(\varepsilon)\right)v.$$

3.2 Algorithms with Guessing Large Items

We now use BranchingMRT to obtain a better approximation ratio. In the new algorithm, we guess the sizes of a few large items in an optimal solution OPT, and then use them to determine the parameter α.

We first remark that, when $|\mathrm{OPT}| \leq 2$, we can easily obtain a 1/2-approximate solution with a single pass. In fact, since $f(\mathrm{OPT}) \leq \sum_{i=1}^{\ell} f(o_i)$ where $\ell = |\mathrm{OPT}|$, at least one of o_i's satisfies $f(o_i) \geq f(\mathrm{OPT})/\ell$, and hence Singleton returns a 1/2-approximate solution when $\ell \leq 2$. Thus, in what follows, we may assume that $|\mathrm{OPT}| \geq 3$.

We start with the case that we have guessed the largest two sizes $c(o_1)$ and $c(o_2)$ in OPT.

▶ **Lemma 12.** *Let $\varepsilon \in (0,1]$, and suppose that $v \leq f(\mathrm{OPT})$ and $c_i \leq c(o_i)/K \leq (1+\varepsilon)c_i$ for $i \in \{1,2\}$. Then, $\tilde{S}' =$ BranchingMRT$(\varepsilon, \alpha, v, c_1, b)$ with $\alpha = 1/(2-c_2)$ or $2/(5-4c_2-c_1)$ and $b = \min\{(1+\varepsilon)c_1, 1/2\}$ satisfies*

$$f(\tilde{S}') \geq \left(\min\left\{\frac{1-c_2}{2-c_2}, \frac{2(1-c_2)}{5-4c_2-c_1}\right\} - O(\varepsilon)\right)v. \tag{5}$$

Proof. Let $\tilde{S} =$ MarginalRatioThresholding(α, v). Note that $f(\tilde{S}') \geq f(\tilde{S})$. If $\tilde{S}$ has size at least $(1-(1+\varepsilon)c_2)K$, then Lemma 3 (1) implies that

$$f(\tilde{S}) \geq \alpha(1-(1+\varepsilon)c_2)v = \alpha(1-c_2)v - O(\varepsilon)v.$$

Otherwise, $c(\tilde{S}) < (1-(1+\varepsilon)c_2)K$. In this case, we see that only the item o_1 can have size more than $(1+\varepsilon)c_2K$, and hence only o_1 can be a bad item. If o_1 is not a bad item, then we have no bad item, and hence Lemma 5 implies that

$$f(\tilde{S}) \geq (1-\alpha)v.$$

If o_1 is bad, then Lemma 10 implies that

$$f(\tilde{S}') \geq \frac{1}{2}\left(1 - \alpha\frac{1-c_1}{2}\right)v - O(\varepsilon)v.$$

Thus the approximation ratio is the minimum of the RHSes of the above three inequalities. This is maximized when $\alpha = 1/(2-c_2)$ or $\alpha = 2/(5-4c_2-c_1)$, and the maximum value is equal to the RHS of (5). ◀

Note that the approximation ratio achieved in Lemma 12 becomes $1/3 - O(\varepsilon)$ when, for example, $c_1 = c_2 = 1/2$. Hence, the above lemma does not show any improvement over Theorem 6 in the worst case. Thus, we next consider the case that we have guessed the largest three sizes $c(o_1)$, $c(o_2)$, and $c(o_3)$ in OPT. Using Lemma 11 in addition to Lemmas 3 (1), 5 and 10, we have the following guarantee.

▶ **Lemma 13.** *Let $\varepsilon \in (0,1]$, and suppose that $v \leq f(\mathrm{OPT})$ and $c_i \leq c(o_i)/K \leq (1+\varepsilon)c_i$ for $i \in \{1,2,3\}$. Then the better output $\tilde{S}'$ of* $\mathsf{BranchingMRT}(\varepsilon, \alpha, v, c_1, b_1)$ *and* $\mathsf{BranchingMRT}(\varepsilon, \alpha, v, c_1, b_2)$ *with $\alpha = 1/(2-c_3)$ or $2/(c_2+3)$, $b_1 = \min\{(1+\varepsilon)c_1, 1/2\}$, and $b_2 = \min\{(1+\varepsilon)c_2, 1/2\}$ satisfies*

$$f(\tilde{S}') \geq \left(\min\left\{\frac{1-c_3}{2-c_3}, \frac{c_2+1}{c_2+3}\right\} - O(\varepsilon)\right) v.$$

Proof. Let $\tilde{S} = \mathsf{MarginalRatioThresholding}(\alpha, v)$. If $\tilde{S}$ has size at least $(1-(1+\varepsilon)c_3)K$, then we have by Lemma 3 (1)

$$f(\tilde{S}) \geq \alpha(1-(1+\varepsilon)c_3)v = \alpha(1-c_3)v - O(\varepsilon)v.$$

Otherwise, $c(\tilde{S}) < (1-(1+\varepsilon)c_3)K$. In this case, we see that only o_1 and o_2 can have size more than $(1+\varepsilon)c_3$, and hence only they can be bad items. If we have no bad item, it holds by Lemma 5 that

$$f(\tilde{S}) \geq (1-\alpha)v.$$

Suppose we have one bad item. If it is o_1 then Lemma 10 with b_1 implies

$$f(\tilde{S}') \geq \left(\frac{1}{2}\left(1-\alpha\frac{1-c_1}{2}\right) - O(\varepsilon)\right) v,$$

and, if it is o_2, we obtain by Lemma 10 with b_2

$$f(\tilde{S}') \geq \left(\frac{1}{2}\left(1-\alpha\frac{1-c_2}{2}\right) - O(\varepsilon)\right) v.$$

Moreover, if we have two bad items o_1 and o_2, then Lemma 11 implies

$$f(\tilde{S}') \geq \left(\frac{1}{3}\left(1+\alpha c_2\right) - O(\varepsilon)\right) v.$$

Therefore, the approximation ratio is the minimum of the RHSes in the above five inequalities, which is maximized to

$$\min\left\{\frac{1-c_3}{2-c_3}, \frac{c_2+1}{c_2+3}\right\} - O(\varepsilon),$$

when $\alpha = 1/(2-c_3)$ or $\alpha = 2/(c_2+3)$. ◀

We now see that we get an approximation ratio of $2/5 - O(\varepsilon)$ by combining the above two lemmas.

▶ **Theorem 14.** *Let $\varepsilon \in (0,1]$ and suppose that $v \leq f(\mathrm{OPT}) \leq (1+\varepsilon)v$ and $c_i \leq c(o_i)/K \leq (1+\varepsilon)c_i$ for $i \in \{1,2,3\}$. If $c(o_1) \leq K/2$, then we can obtain a $(2/5 - O(\varepsilon))$-approximate solution with a single pass.*

Proof. We run the two algorithms with the optimal α shown in Lemmas 12 and 13 in parallel. Let $\tilde{S}$ be the output with the better function value. Then, we have $f(\tilde{S}) \geq \beta v$, where

$$\beta = \max\left\{\min\left\{\frac{1-c_2}{2-c_2}, \frac{2(1-c_2)}{5-4c_2-c_1}\right\}, \min\left\{\frac{1-c_3}{2-c_3}, \frac{c_2+1}{c_2+3}\right\}\right\} - O(\varepsilon).$$

We can confirm that the first term is at least $2/5$, and thus $\tilde{S}$ is a $(2/5 - O(\varepsilon))$-approximate solution. ◀

Algorithm 5

```
procedure DynamicBranchingMRT($\varepsilon$)
    $\mathcal{V} := \{(1+\varepsilon)^i \mid i \in \mathbb{Z}_+\}$.
    For each $c_1, c_2, c_3 \in \mathcal{V}$ with $c_3 \le c_2 \le c_1 \le 1/2$ and each $b \in \{(1+\varepsilon)c_1, (1+\varepsilon)c_2, 1/2\}$,
do the following with $\alpha$ defined based on Lemmas 12 and 13.
    For each $v \in \mathcal{V}$, set $S_v := \emptyset$.
    while item $e$ is arriving do
        Delete $e$ with $c(e) > (1+\varepsilon)c_1 K$.
        $m := \max\{m, f(e)\}$.
        $\mathcal{I} := \{v \in \mathcal{V} \mid m \le v \le Km/\alpha\}$.
        Delete $S_v$ (along with $\hat{S}_v$ and $S'_v$ if exists) such that $v \notin \mathcal{I}$.
        for $v \in \mathcal{V}$ do
            if $f(S_v) < \lambda$ then
                if $\frac{f(e|S_v)}{c(e)} \ge \frac{\alpha v - f(S_v)}{K - c(S_v)}$ and $c(S_v + e) \le K$ then $S_v := S_v + e$.
                if $f(S_v) \ge \lambda$ then
                    if $c(S) \ge (1-b)K$ then $S' := \{e\}$ else $S' := S$.
                    $\hat{S}_v := S'$.
            else
                if $\frac{f(e|S_v)}{c(e)} \ge \frac{\alpha v - f(S_v)}{K - c(S_v)}$ and $c(S_v + e) \le K$ then $S_v := S_v + e$.
                if $\frac{f(e|S_v)}{c(e)} \ge \frac{\alpha v - f(S_v)}{K - c(S_v)}$ and $c(S_v + e) > K$ then
                    if $f(S'_v) < f(\hat{S}_v + e)$ then $S'_v := \hat{S}_v + e$.
    $S := S_v$ for $v \in \mathcal{I}$ that maximizes $f(S_v)$.
    $S' := S'_v$ for $v \in \mathcal{I}$ that maximizes $f(S'_v)$.
    return $S$ or $S'$ whichever has the larger function value.
```

To eliminate the assumption that we are given v, we can design a dynamic-update version of BranchingMRT by keeping the interval that contains the optimal value, similarly to Theorem 7. DynamicBranchingMRT, given in Algorithm 5, is a dynamic-update version of BranchingMRT. The proof for updating the interval $\mathcal{I}$ dynamically is the same as the proof of Theorem 7. The number of streams for guessing v is $O(\log(K)/\varepsilon)$. We also guess c_i for $i \in \{1,2,3\}$ from $\{(1+\varepsilon)^j \mid j \in \mathbb{Z}_+\}$. As $1 \le c(o_i) \le K/2$ for $i \in \{1,2,3\}$, the number of guessing for c_i is $O(\log(K)/\varepsilon)$. Hence, including v, there are $O((\log(K)/\varepsilon)^4)$ streams in parallel. To summarize, we obtain the following:

▶ **Theorem 15.** *Suppose that $c(o_1) \le K/2$. The algorithm that runs* DynamicBranchingMRT *and* Singleton *in parallel and takes the better output is a $(2/5-\varepsilon)$-approximation streaming algorithm with a single pass for the problem* (1). *The space complexity of the algorithm is $O(K(\log(K)/\varepsilon)^4)$.*

4 Single-Pass $(4/11 - \varepsilon)$-Approximation Algorithm

In this section, we consider the case that $c(o_1)$ is larger than $K/2$. For the purpose, we consider the problem of finding a set S of items that maximizes $f(S)$ subject to the constraint that the total size is at most pK, for a given number $p \ge 2$. We say that a set S of items is a (p,α)-*approximate solution* if $c(S) \le pK$ and $f(S) \ge \alpha f(\mathrm{OPT})$, where OPT is an optimal solution of the original instance.

▶ **Theorem 16.** *For a number $p \geq 2$, there is a $\left(p, \frac{2p}{2p+3} - \varepsilon\right)$-approximation streaming algorithm with a single pass for the problem* (1). *In particular, when $p = 2$, it admits $(2, 4/7 - \varepsilon)$-approximation. The space complexity of the algorithm is $O(K(\log(K)/\varepsilon)^3)$.*

The basic framework of the algorithm is the same as in Section 3; we design a thresholding algorithm and a branching algorithm, where the parameters are different and the analysis is simpler.

Using Theorem 16, we can design a $(4/11 - \varepsilon)$-approximation streaming algorithm for an instance having a large item.

▶ **Theorem 17.** *For the problem* (1), *there exists a $(4/11 - \varepsilon)$-approximation streaming algorithm with a single pass. The space complexity of the algorithm is $O(K(\log(K)/\varepsilon)^4)$.*

Proof. Let o_1 be an item in OPT with the maximum size. If $c(o_1) \leq K/2$, then Theorem 15 gives a $(2/5 - O(\varepsilon))$-approximate solution, and thus we may assume that $c(o_1) > K/2$. Note that there exists only one item whose size is more than $K/2$. Let β be the target approximation ratio which will be determined later. We may assume that $f(o_1) < \beta f(\mathrm{OPT})$, as otherwise Singleton (Algorithm 2) gives β-approximation. Then, we see $f(\mathrm{OPT} - o_1) > (1 - \beta) f(\mathrm{OPT})$ and $c(\mathrm{OPT} - o_1) < K/2$. Consider maximizing $f(S)$ subject to $c(S) \leq K/2$ in the set $\{e \in E \mid c(e) \leq K/2\}$. The optimal value is at least $f(\mathrm{OPT} - o_1) > (1 - \beta) f(\mathrm{OPT})$. We now apply Theorem 16 with $p = 2$ to this problem. Then, the output $\tilde{S}$ has size at most K, and moreover, we have $f(\tilde{S}) \geq \left(\frac{4}{7} - O(\varepsilon)\right)(1 - \beta) f(\mathrm{OPT})$. Thus, we obtain $\min\{\beta, (\frac{4}{7} - O(\varepsilon))(1 - \beta)\}$-approximation. This approximation ratio is maximized to $4/11$ when $\beta = 4/11$. ◀

5 Multiple-Pass Streaming Algorithm

In this section, we provide a multiple-pass streaming algorithm with approximation ratio $2/5 - \varepsilon$.

We first consider a generalization of the original problem. Let $E_R \subseteq E$ be a subset of the ground set E. For ease of presentation, we will call E_R the *red* items. Consider the problem defined below:

$$\text{maximize} \quad f(S) \quad \text{subject to} \quad c(S) \leq K, \quad |S \cap E_R| \leq 1. \tag{6}$$

In the following, we show that, given $\varepsilon \in (0, 1]$, an approximation v to $f(\mathrm{OPT})$ with $v \leq f(\mathrm{OPT}) \leq (1 + \varepsilon)v$, and an approximation θ to $f(o_\mathrm{r})$ for the unique item o_r in $\mathrm{OPT} \cap E_R$, we can choose $O(1)$ of the red items so that one of them $e \in E_R$ satisfies that $f(\mathrm{OPT} - o_\mathrm{r} + e) \geq (\Gamma(\theta) - O(\varepsilon))v$, where $\Gamma(\cdot)$ is a piecewise linear function lower-bounded by $2/3$. For technical reasons, we will choose θ to be one of the geometric series $(1 + \varepsilon)^i/2$ for $i \in \mathbb{Z}$.

▶ **Theorem 18.** *Suppose that we are given $\varepsilon \in (0, 1]$, $v \in \mathbb{R}_+$ with $v \leq f(\mathrm{OPT}) \leq (1 + \varepsilon)v$, and $\theta \in \mathbb{R}_+$ with the following property:*
1. *if $\theta \leq 1/2$, $\theta v/(1 + \varepsilon) \leq f(o_\mathrm{r}) \leq \theta v$,*
2. *if $\theta \geq 1/2$, $\theta v \leq f(o_\mathrm{r}) \leq (1 + \varepsilon)\theta v \leq v$.*

Then, there is a single-pass streaming algorithm that chooses a set S of red items in E_R with constant size such that (i) *for any item $e \in S$, $\theta v/(1 + \varepsilon) \leq f(o_\mathrm{r}) \leq \theta v$ when $\theta \leq 1/2$ and $\theta v \leq f(o_\mathrm{r}) \leq (1 + \varepsilon)\theta v \leq v$ when $\theta \geq 1/2$, and* (ii) *some item $e \in S$ satisfies that $f(\mathrm{OPT} - o_\mathrm{r} + e) \geq (\Gamma(\theta) - O(\varepsilon))v$, where $\Gamma(\theta)$ is defined as follows: when $\theta \in (0, 1/2)$,*

$$\Gamma(\theta) = \max\left\{\frac{t(t + 3)}{(t + 1)(t + 2)} - \frac{t - 1}{t + 1}\theta \mid t \in \mathbb{Z}_+, t > \frac{1}{\theta} - 2\right\}, \tag{7}$$

when $\theta \in [1/2, 2/3)$, $\Gamma(\theta) = 2/3$, and when $\theta \in [2/3, 1]$, $\Gamma(\theta) = \theta$.

Algorithm 6

```
procedure MultiPassKnapsack($\varepsilon, v, \theta, c_1$)    ▷ $\varepsilon \in (0,1]$, $v \in \mathbb{R}_+$, and $\theta, c_1 \in [0,1]$.
    Use the algorithm in Theorem 18 to choose a set $S$ of items $e$ with $c_1/(1+\varepsilon) \leq c(e)/K \leq c_1$ so that one of them $e \in S$ satisfies $f(\mathrm{OPT} - o_1 + e) \geq v(\Gamma(\theta) - O(\varepsilon))$.
    for each item $e \in S$ do
        Define a submodular function $g_e(\cdot) = f(\cdot \mid e)$.
        Apply the marginal-ratio thresholding algorithm (Lemma 21) with regard to function $g_e$, where $h = \frac{1-c_1}{1-(c_1/(1+\varepsilon))}$ and $K' = (1-(c_1/(1+\varepsilon))K$.
        Let the resultant set be $S_e$.
    return the solution $S_e \cup \{e\}$ with $\max_{e \in S} f(S_e + e)$.
```

We next show that when $c(o_1) \geq K/2$, we can use multiple passes to get a $(2/5 - \varepsilon)$-approximation for the problem (1). Let $\mathrm{OPT} = \{o_1, o_2, \ldots, o_\ell\}$ be an optimal solution with $c(o_1) \geq c(o_2) \geq \cdots \geq c(o_\ell)$. Suppose that $c_1 \in \mathbb{R}_+$ satisfies $1/2 \leq c_1/(1+\varepsilon) \leq c(o_1)/K \leq c_1$. We observe the following claims.

▶ **Claim 19.** *When $c(o_1) \geq K/2$, we may assume that $\frac{3}{10} f(\mathrm{OPT}) < f(o_1) < \frac{2}{5} f(\mathrm{OPT})$.*

▶ **Claim 20.** *We may assume that $c(o_1) \leq (1+\varepsilon)\frac{2}{3}K$.*

We use the first pass to estimate $f(\mathrm{OPT})$ as follows. For an error parameter $\varepsilon \in (0,1]$, perform the single-pass algorithm in Theorem 7 to get a $(1/3-\varepsilon)$-approximate solution $S \subseteq E$, which can be used to upper bound the value of $f(\mathrm{OPT})$, that is, $f(S) \leq f(\mathrm{OPT}) \leq (3+\varepsilon) f(S)$. We then find the geometric series to guess its exact value. Thus, we may assume that we are given the value v with $v \leq f(\mathrm{OPT}) \leq (1+\varepsilon)v$.

Below we show how to obtain a solution of value at least $(2/5 - O(\varepsilon))v$, using two more passes. Before we start, we introduce a slightly modified versions of the algorithms presented in Section 2; it will be used as a subroutine.

▶ **Lemma 21.** *Consider the problem (1) with the knapsack capacity K'. Let $h \in \mathbb{R}_+$, and suppose that Algorithms 1 and 2 are modified as follows:*

- *(At Line 4 in Algorithm 1) A new item e is added into the current set S only if $\frac{f(e|S)}{c(e)} \geq \frac{\alpha v - f(S)}{hK' - c(S)}$ and $c(S+e) \leq hK'$.*
- *(At Line 4 in Algorithm 2) A new item e is taken into account only if $c(e) \leq hK'$.*

Then, the best returned set $\tilde{S}$ of the two algorithms with $\alpha = \frac{2h}{h+2}$ satisfies that $c(\tilde{S}) \leq hK'$ and $f(\tilde{S}) \geq \frac{h}{h+2} v$. Moreover, we can obtain a $\left(\frac{h}{h+2} - O(\varepsilon)\right)$-approximate solution with the dynamic update technique.

Let all items $e \in E$ whose sizes $c(e)$ satisfy $c_1/(1+\varepsilon) \leq c(e)/K \leq c_1$ be the red items. By Theorem 18, we can select a set S of the red items so that one of them guarantees $f(\mathrm{OPT} - o_1 + e) \geq (\Gamma(\theta) - O(\varepsilon))v$, where θ satisfies the condition in Theorem 18. Note that any $e \in S$ satisfies $f(e) \geq \theta v/(1+\varepsilon)$. Also, by Claim 19, we see $\frac{3}{10} v < \theta < \frac{2}{5}(1+\varepsilon)v$.

In the next pass, for each $e \in S$, define a new monotone submodular function $g_e(\cdot) = f(\cdot \mid e)$ and apply the modified thresholding algorithm (Lemma 21) with $h = \frac{1-c_1}{1-(c_1/(1+\varepsilon))}$ and $K' = (1-(c_1/(1+\varepsilon))K$. Let S_e be the output of the modified thresholding algorithm. Then our algorithm returns the solution $S_e \cup \{e\}$ with $\max_{e \in S} f(S_e + e)$. The detail is given in Algorithm 6.

The returned solution has size at most K, since $c(S_e) \leq (1-c_1)K$ by Lemma 21. Moreover, it follows that the returned solution $\tilde{S}$ satisfies that $f(\tilde{S}) \geq (2/5 - O(\varepsilon))v$. The next theorem summarizes our results in this section.

▶ **Theorem 22.** *For $\varepsilon \in (0, 1]$, suppose that $v \le f(\mathrm{OPT}) \le (1+\varepsilon)v$, $1/2 \le c_1/(1+\varepsilon) \le c(o_1)/K \le c_1$, and θ satisfies the condition in Theorem 18. After running* MultiPassKnapsack$(\varepsilon, v, \theta, c_1)$*, there exists an item $e \in S$ chosen in Line 2, which, along with S_e collected in Line 6, gives $f(S_e + e) \ge (2/5 - O(\varepsilon))v$.*

▶ **Theorem 23.** *Suppose that $c(o_1) > K/2$. There exists an algorithm that uses MultiPassKnapsack as a subroutine so that it returns $(2/5 - \varepsilon)$-approximation with 3 passes for the problem (1). The space complexity of the algorithm is $O(K(\log(K)/\varepsilon)^2)$.*

References

1 Noga Alon, Iftah Gamzu, and Moshe Tennenholtz. Optimizing budget allocation among channels and influencers. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*, pages 381–388, 2012.

2 Ashwinkumar Badanidiyuru, Baharan Mirzasoleiman, Amin Karbasi, and Andreas Krause. Streaming submodular maximization: massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 671–680, 2014.

3 Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1497–1514, 2013.

4 Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

5 Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: matchings, matroids, and more. *Mathematical Programming*, 154(1-2):225–247, 2015. `doi:10.1007/s10107-015-0900-7`.

6 T.-H. Hubert Chan, Zhiyi Huang, Shaofeng H.-C. Jiang, Ning Kang, and Zhihao Gavin Tang. Online submodular maximization with free disposal: Randomization beats for partition matroids online. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1204–1223, 2017.

7 Chandra Chekuri, Shalmoli Gupta, and Kent Quanrud. Streaming algorithms for submodular function maximization. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 9134, pages 318–330, 2015.

8 Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM Journal on Computing*, 43(6):1831–1879, 2014. `doi:10.1137/110839655`.

9 Yuval Filmus and Justin Ward. A tight combinatorial algorithm for submodular maximization subject to a matroid constraint. *SIAM Journal on Computing*, 43(2):514–542, 2014.

10 M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions ii. *Mathematical Programming Study*, 8:73–87, 1978.

11 David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 137–146, 2003.

12 Andreas Krause, Ajit Paul Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

13 Ariel Kulik, Hadas Shachnai, and Tami Tamir. Maximizing submodular set functions subject to multiple linear constraints. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 545–554, 2013.

14 Jon Lee. *Maximum Entropy Sampling*, volume 3 of *Encyclopedia of Environmetrics*, pages 1229–1234. John Wiley & Sons, Ltd., 2006.

15 Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Mathematics of Operations Research*, 35(4):795–806, 2010.

16 Hui Lin and Jeff Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 912–920, 2010.

17 Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT)*, pages 510–520, 2011.

18 Tasuku Soma, Naonori Kakimura, Kazuhiro Inaba, and Ken-ichi Kawarabayashi. Optimal budget allocation: Theoretical guarantee and efficient algorithm. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 351–359, 2014.

19 Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters*, 32(1):41–43, 2004.

20 Laurence Wolsey. Maximising real-valued submodular functions: primal and dual heuristics for location problems. *Mathematics of Operations Research*, 1982.

21 Qilian Yu, Easton Li Xu, and Shuguang Cui. Streaming algorithms for news and scientific literature recommendation: Submodular maximization with a d-knapsack constraint. *IEEE Global Conference on Signal and Information Processing*, 2016.

Fractional Set Cover in the Streaming Model*

Piotr Indyk[1], Sepideh Mahabadi[2], Ronitt Rubinfeld[3], Jonathan Ullman[4], Ali Vakilian[5], and Anak Yodpinyanee[6]

1 **CSAIL, MIT, Cambridge, MA, USA**
`indyk@mit.edu`
2 **CSAIL, MIT, Cambridge, MA, USA**
`mahabadi@mit.edu`
3 **CSAIL, MIT and TAU, Cambridge, MA, USA**
`ronitt@csail.mit.edu`
4 **CCIS, Northeastern University, Boston, MA, USA**
`jullman@ccs.neu.edu`
5 **CSAIL, MIT, Cambridge, MA, USA**
`vakilian@mit.edu`
6 **CSAIL, MIT, Cambridge, MA, USA**
`anak@mit.edu`

Abstract

We study the *Fractional Set Cover* problem in the streaming model. That is, we consider the relaxation of the set cover problem over a universe of n elements and a collection of m sets, where each set can be picked fractionally, with a value in $[0, 1]$. We present a randomized $(1+\varepsilon)$-approximation algorithm that makes p passes over the data, and uses $\widetilde{O}(mn^{O(1/p\varepsilon)} + n)$ memory space. The algorithm works in both the set arrival and the edge arrival models. To the best of our knowledge, this is the first streaming result for the fractional set cover problem. We obtain our results by employing the multiplicative weights update framework in the streaming settings.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Streaming Algorithms, Fractional Set Cover, LP relaxation, Multiplicative Weight Update

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.12

1 Introduction

Set Cover is one of the classical NP-hard problems in combinatorial optimization. In this problem the input consists of a set (universe) of n elements $\mathcal{U} = \{e_1, \cdots, e_n\}$ and a collection of m sets $\mathcal{F} = \{S_1, \cdots, S_m\}$. The goal is to find the minimum size *set cover* of $\mathcal{U}$, i.e., a collection of sets in $\mathcal{F}$ whose union is $\mathcal{U}$. The LP relaxation of Set Cover (called SetCover-LP) is also well-studied. It is a continuous relaxation of the problem where each set $S \in \mathcal{F}$ can be selected “fractionally”, i.e., assigned a number x_S from $[0, 1]$, such that for each element e its “fractional coverage” $\sum_{S:e\in S} x_S$ is at least 1, and the sum $\sum_S x_S$ is minimized. Both variants are well-studied and have many applications in operations research [23, 25, 11], information retrieval and data mining [34], learning theory [26], web host analysis [15], etc.

* PI, SM and AV are supported by grants from the NSF and the Simons Investigator award. RR is supported by the Israel Science Foundation (ISF) grant 1536/14, the NSF grants CCF-1420692 and CCF-1650733. AY is supprted by the NSF grants CCF-1420692, CCF-1650733, CCF-1065125, and the DPST scholarship, Royal Thai Government.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 12; pp. 12:1–12:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

A natural $\ln n$-approximation greedy algorithm of Set Cover, which in each iteration picks the *best* remaining set, is widely used and known to be the best possible under $\mathbf{P} \neq \mathbf{NP}$ [29, 21, 33, 5, 31, 18]. However, the greedy algorithm is sequential in nature and does not perform efficiently in the standard models developed for *massive data analysis*; in particular, in the *streaming* model. In streaming Set Cover [34], the ground set $\mathcal{U}$ is stored in the memory, the sets $S_1, \cdots, S_m$ are stored consecutively in a read-only repository and the algorithm can only access the sets by performing sequential scans (or passes) over the repository. Moreover, the amount of (read-write) memory available to the algorithm is much smaller than the input size (which can be as large as mn). The objective is to design a *space-efficient* algorithm that returns a (nearly)-optimal feasible cover of $\mathcal{U}$ after performing only a few passes over the data. Streaming Set Cover has witnessed a lot of developments in recent years, and tight upper and lower bounds are known, in both *low space* [20, 13] and *low approximation* [17, 24, 8, 12, 7] regimes.

Despite the above developments, the results for the *fractional* variant of the problem are still unsatisfactory. To the best of our knowledge, it is not known whether there exists an efficient and accurate algorithm for this problem that uses only a logarithmic (or even a *poly logarithmic*) number of passes. This state of affairs is perhaps surprising, given the many recent developments on fast LP solvers [27, 37, 28, 4, 3, 35]. To the best of our knowledge, the only prior results on streaming Packing/Covering LPs were presented in paper [1], which studied the LP relaxation of Maximum Matching.

1.1 Our Results

In this paper, we present the first $(1+\varepsilon)$-approximation algorithm for the fractional Set Cover in the streaming model with constant number of passes. Our algorithm performs p passes over the data stream and uses $\widetilde{O}(mn^{O(\frac{1}{p\varepsilon})}+n)$ memory space to return a $(1+\varepsilon)$ approximate solution of the LP relaxation of Set Cover for positive parameter $\varepsilon \leq 1/2$.

We emphasize that similarly to the previous work on variants of Set Cover in streaming setting, our result also holds for the *edge arrival* stream in which the pair of (S_i, e_j) (edges) are stored in the read-only repository and all elements of a set are not necessarily stored consecutively.

1.2 Related work

Set Cover Problem. The Set Cover problem was first studied in the streaming model in [34], which presented an $O(\log n)$-approximation algorithm in $O(\log n)$ passes and using $\widetilde{O}(n)$ space. This approximation factor and the number of passes can be improved to $O(\log n)$ by adapting the greedy algorithm *thresholding* idea presented in [16] . In the low space regime ($\widetilde{O}(n)$ space), Emek and Rosen [20] designed a *deterministic* single pass algorithm that achieves an $O(\sqrt{n})$-approximation. This is provably the best guarantee that one can hope for in a single pass even considering randomized algorithms. Later Chakrabarti and Wirth [13] generalized this result and provided a *tight* trade-off bounds for Set Cover in multiple passes. More precisely, they gave an $O(pn^{1/(p+1)})$-approximate algorithm in p-passes using $\widetilde{O}(n)$ space and proved that this is the best possible approximation ratio up to a factor of poly(p) in p passes and $\widetilde{O}(n)$ space.

A different line of work started by Demaine et al. [17] focused on designing a "low" approximation algorithm (between $\Theta(1)$ and $\Theta(\log n)$) in the smallest possible amount of space. In contrast to the results in the $\widetilde{O}(n)$ space regime, [17] showed that randomness is necessary: any constant pass deterministic algorithm requires $\Omega(mn)$ space to achieve constant approximation guarantee. Further, they provided a $O(4^p \log n)$-approximation algorithm that makes $O(4^p)$ passes and uses $\widetilde{O}(mn^{1/p}+n)$. Later Har-Peled et al. [24] improved the algorithm to a $2p$-pass $O(p \log n)$-approximation with memory space $\widetilde{O}(mn^{1/p}+n)$[1]. The result was further improved by Bateni et al. where they designed a p-pass algorithm that returns a $(1+\varepsilon)\log n$-approximate solution using $mn^{\Theta(1/p)}$ memory [12].

As for the lower bounds, Assadi et al. [8] presented a lower bound of $\Omega(mn/\alpha)$ memory for any single pass streaming algorithm that computes a α-approxime solution. For the problem of estimating the size of an optimal solution they prove $\Omega(mn/\alpha^2)$ memory lower bound. For both settings, they complement the results with matching tight upper bounds. Very recently, Assadi [7] proved a lower bound for streaming algorithms with multiple passes which is tight up to polylog factors: any α-approximation algorithm for Set Cover requires $\Omega(mn^{1/\alpha})$ space, even if it is allowed polylog(n) passes over the stream, and even if the sets are arriving in a random order in the stream. Further, [7] provided the matching upper bound: a $(2\alpha+1)$-pass algorithm that computes a $(\alpha+\varepsilon)$-approximate solution in $\widetilde{O}(\frac{mn^{1/\alpha}}{\varepsilon^2}+\frac{n}{\varepsilon})$ memory (assuming exponential computational resource).

Max Cover Problem. The first result on streaming Max k-Cover showed how to compute a $(1/4)$-approximate solution in one pass using $\widetilde{O}(kn)$ space [34]. It was improved by Badanidiyuru et al. [9] to a $(1/2-\varepsilon)$-approximation algorithm that requires $\widetilde{O}(n/\varepsilon)$ space. Moreover, their algorithm works for a more general problem of Submodular Maximization with cardinality constraints. This result was later generalized for the problem of non-monotone submodular maximization under constraints beyond cardinality [14]. Recently, McGregor and Vu [30] and Bateni et al. [12] independently obtained single pass $(1-1/e-\varepsilon)$-approximation with $\widetilde{O}(m/\varepsilon^2)$ space. On the lower bound side, [30] showed a lower bound of $\widetilde{\Omega}(m)$ for constant pass algorithm whose approximation is better than $(1-1/e)$. Moreover, [7] proved that any streaming $(1-\varepsilon)$-approximation algorithm of Max k-Cover in polylog(n) passes requires $\widetilde{\Omega}(m/\varepsilon^2)$ space even on random order streams and the case $k=O(1)$. This bound is also complemented by the $\widetilde{O}(mk/\varepsilon^2)$ and $\widetilde{O}(m/\varepsilon^3)$ algorithms of [12, 30]. For more detailed survey of the results on streaming Max k-Cover refer to [12, 30, 7].

Covering/Packing LPs. The study of LPs in streaming model was first discussed in the work of Ahn and Guha [1] where they used *multiplicative weights update* (MWU) based techniques to solve the LP relaxation of Maximum (Weighted) Matching problem. They used the fact that MWU returns a near optimal fractional solution with small size support: first they solve the fractional matching problem, then solve the actual matching only considering the edges in the support of the returned fractional solution.

Our algorithm is also based on the MWU method, which is one of the main key techniques in designing fast approximation algorithms for Covering and Packing LPs [32, 36, 22, 6]. We note that the MWU method has been previously studied in the context of *streaming* and *distributed* algorithms, leading to efficient algorithms for a wide range of graph optimization problems [1, 10, 2].

For a related problem, *covering integer LP* (covering ILP), Assadi et al. [8] designed a one pass streaming algorithm that estimates the optimal solution of $\{\min \mathbf{c}^\top\mathbf{x} \mid \mathbf{A}^\top\mathbf{x} \geq \mathbf{b}, \mathbf{x} \in \{0,1\}^n\}$ within a factor of α using $\widetilde{O}(\frac{mn}{\alpha^2}\cdot b_{\max}+m+n\cdot b_{\max})$ where $b_{\max}$ denotes

[1] In streaming model, space complexity is of interest and one can assume exponentital computation power. In this case the algorithms of [17, 24] save a factor of $\log n$ in the approximation ratio.

the largest entry of $\mathbf{b}$. In this problem, they assume that columns of $\mathbf{A}$, constriants, are given one by one in the stream.

In a different regime, [19] studied approximating the feasibility LP in streaming model with additive approximation. Their algorithm performs two passes and is most efficient when the input is dense.

1.3 Our Techniques

Preprocessing. Let k denote the value of the optimal solution. The algorithm starts by picking a uniform *fractional* vector (each entry of value $O(\frac{k}{m})$) which covers all frequently occurring elements (those appearing in $\Omega(\frac{m}{k})$ sets), and updates the uncovered elements in one pass. This step considerably reduces the memory usage as the uncovered elements have now lower occurrence (roughly $\frac{m}{k}$). Note that we do not need to assume the knowledge of the correct value k: in parallel we try all powers of $(1+\varepsilon)$, denoting our guess by ℓ.

Multiplicative Weight Update. To cover the remaining elements, we employ the MWU framework and show how to implement it in the streaming setting. In each iteration of MWU, we have a probability distribution $\mathbf{p}$ corresponding to the constraints (elements) and we need to satisfy the *average* covering constraint. More precisely, we need an *oracle* that assigns values to x_S for each set S so that $\sum_S p_S x_S \geq 1$ subject to $\|\mathbf{x}\|_1 \leq \ell$, where p_S is the sum of probabilities of the elements in the set S. Then, the algorithm needs to update $\mathbf{p}$ according to the amount each element has been covered by the oracle's solution. The simple greedy realization of the oracle can be implemented in the streaming setting efficiently by computing all p_S while reading the stream in one pass, then choosing the heaviest set (i.e., the set with largest p_S) and setting its x_S to ℓ. This approach works, except that the number of rounds T required by the MWU framework is large. In fact, $T = \Omega(\frac{\phi \log n}{\varepsilon^2})$, where ϕ is the width parameter (the maximum amount an oracle solution may over-cover an element), which is $\Theta(\ell)$ in this naïve realization. Next, we show how to decrease T in two steps.

Step 1. A first hope would be that there is a more efficient implementation of the oracle which gives a better width parameter. Nonetheless, no matter how the oracle is implemented, if all sets in $\mathcal{F}$ contain a fixed element e, then the width is inevitably $\Omega(\ell)$. This observation implies that we need to work with a different set system that has small width, but at the same time, it has the same objective value as of the optimal solution. Consequently, we consider the *extended set system* where we replace $\mathcal{F}$ with all subsets of the sets in $\mathcal{F}$. This extended system preserves the optimality, and under this system we may avoid over-covering elements and obtain $T = O(\log n)$ (for constant ε).

In order to turn a solution in our set system into a solution in the extended set system with small width, we need to remove the repeated elements from the sets in the solution so that every covered element appears exactly once, and thereby getting constant width. However, as a side effect, this reduces the total weight of the solution ($\sum_{S \in \text{SOL}} p_S x_S$), and thus the average covering constraint might not be satisfied anymore. In fact, we need to come up with a guarantee that, on one hand, is preserved under the pruning step, and on the other hand, implies that the solution has large enough total weight

Therefore, to fulfill the average constraint under the pruning step, the oracle must instead solve the *maximum coverage* problem: given a budget, choose sets to cover the largest (fractional) amount of elements. We first show that this problem can be solved approximately via the MWU framework using the simple oracle that picks the heaviest set, but this MWU algorithm still requires T passes over the data. To improve the number of

SetCover-LP $\langle\langle$*Input:* $\mathcal{U}, \mathcal{F}\rangle\rangle$

$$\begin{array}{lll} \text{minimize} & \sum_{S \in \mathcal{F}} x_S & \\ \text{subject to} & \sum_{S: e \in S} x_S \geq 1 & \forall e \in \mathcal{U} \\ & x_S \geq 0 & \forall S \in \mathcal{F} \end{array}$$

Figure 1 LP relaxation of Set Cover.

passes, we perform *element sampling* and apply the MWU algorithm to find an approximate maximum coverage of a small number of sampled elements, whose subproblem can be stored in memory. Fortunately, while the number of fractional solutions to maximum coverage is unbounded, by exploiting the structure of the solutions returned by the MWU method, we can limit the number of plausible solutions of this oracle and approximately solve the average constraint, thereby reducing the space usage to $\widetilde{O}(m)$ for a $O(\frac{\log n}{\varepsilon^2})$-pass algorithm.

Step 2. To further reduce the number of required passes, we observe that the weights of the constraints change slowly. Thus, in a single pass, we can sample the elements for multiple rounds in advance, and then perform rejection (sub-)sampling to obtain an unbiased set of samples for each subsequent round. This will lead to a streaming algorithm with p passes and $mn^{O(1/p)}$ space.

Extension. We also extend our result to handle general covering LPs. More specifically, in the LP relaxation of Set Cover, maximize $\mathbf{c}^\top \mathbf{x}$ subject to $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ and $\mathbf{x} \geq \mathbf{0}$, $\mathbf{A}$ has entries from $\{0, 1\}$ whereas entries of $\mathbf{b}$ and $\mathbf{c}$ are all ones. If the non-zero entries instead belong to a range $[1, M]$, we increase the number of sampled elements by poly(M) to handle discrepancies between coefficients, leading to a poly(M)-multiplicative overhead in the space usage.

2 MWU Framework of the Streaming Algorithm for Fractional Set Cover

In this section, we present a basic streaming algorithm that computes a $(1+\varepsilon)$-approximate solution of the LP-relaxation of Set Cover for any $\varepsilon > 0$ via the MWU framework. We will, in the next section, improve it into an efficient algorithm that achieves the claimed $O(p)$ passes and $\widetilde{O}(mn^{1/p})$ space complexity.

Let $\mathcal{U}$ and $\mathcal{F}$ be the ground set of elements and the collection of sets, respectively, and recall that $|\mathcal{U}| = n$ and $|\mathcal{F}| = m$. Let $\mathbf{x} \in \mathbb{R}^m$ be a vector indexed by the sets in $\mathcal{F}$, where x_S denotes the value assigned to the set S. Our goal is to compute an approximate solution to the LP in Figure 1. Throughout the analysis we assume $\varepsilon \leq 1/2$, and ignore the case where some element never appears in any set, as it is easy to detect in a single pass that no cover is valid. For ease of reading, we write $\widetilde{O}$ and $\widetilde{\Theta}$ to hide polylog$(m, n, \frac{1}{\varepsilon})$ factors.

Outline of the algorithm. Let k denote the optimal objective value, and $0 < \varepsilon \leq 1/2$ be a parameter. The outline of the algorithm is shown in **fracSetCover** (Figure 2). This algorithm makes calls to the subroutine **feasibilityTest**, that given a parameter ℓ, with high probability, either returns a solution of objective value at most $(1+\varepsilon/3)\ell$, or detects that the optimal objective value exceeds ℓ. Consequently, we may search for the right value of ℓ by

```
fracSetCover($\varepsilon$):
  ▷ Finds a feasible $(1+\varepsilon)$-approximate solution in $O(\frac{\log n}{\varepsilon})$ iterations
  for $\ell \in \{(1+\varepsilon/3)^i \mid 0 \le i \le \log_{1+\varepsilon/3} n\}$ do in parallel: $\mathbf{x}_\ell \leftarrow$ feasibilityTest$(\ell, \varepsilon/3)$
  return $x_{\ell^*}$ where $\ell^* \leftarrow \min\{\ell : x_\ell$ is not INFEASIBLE$\}$
```

Figure 2 **fracSetCover** returns a $(1+\varepsilon)$-approximate solution of $SetCover - LP$, where **feasibilityTest** is an algorithm that returns a solution of objective value at most $(1+\varepsilon/3)\ell$ when $\ell \ge k$.

considering all values in $\{(1+\varepsilon/3)^i \mid 0 \le i \le \log_{1+\varepsilon/3} n\}$. As for some value of ℓ it holds that $k \le \ell \le k(1+\varepsilon/3)$, we obtain a solution of size $(1+\varepsilon/3)\ell \le (1+\varepsilon/3)(1+\varepsilon/3)k \le (1+\varepsilon)k$ which gives an approximation factor $(1+\varepsilon)$. This whole process of searching for k increases the space complexity of the algorithm by at most a multiplicative factor of $\log_{1+\varepsilon/3} n \approx \frac{3\log n}{\varepsilon}$.

The **feasibilityTest** subroutine employs the multiplicative weights update method (MWU) which is described next.

2.1 Preliminaries of the MWU method for solving covering LPs

In the following, we describe the MWU framework. The claims presented here are standard results of the MWU method. For more details, see e.g. Section 3 of [6]. Note that we introduce the general LP notation as it simplifies the presentation later on.

Let $\mathbf{A}\mathbf{x} \ge \mathbf{b}$ be a set of linear constraints, and let $\mathcal{P} \triangleq \{\mathbf{x} \in \mathbb{R}^m : \mathbf{x} \ge \mathbf{0}\}$ be the polytope of the non-negative orthant. For a given error parameter $0 < \beta < 1$, we would like to solve an approximate version of the feasibility problem by doing one of the following:

- Compute $\hat{\mathbf{x}} \in \mathcal{P}$ such that $\mathbf{A}_i\hat{\mathbf{x}} - b_i \ge -\beta$ for *every* constraint i.
- Correctly report that the system $\mathbf{A}\mathbf{x} \ge \mathbf{b}$ has no solution in $\mathcal{P}$.

The MWU method solves this problem assuming the existence of the following oracle that takes a distribution $\mathbf{p}$ over the constraints and finds a solution $\hat{\mathbf{x}}$ that satisfies the constraints on average over $\mathbf{p}$.

▶ **Definition 2.1.** Let $\phi \ge 1$ be a width parameter and $0 < \beta < 1$ be an error parameter. A $(1,\phi)$*-bounded* $(\beta/3)$*-approximate oracle* is an algorithm that takes as input a distribution $\mathbf{p}$ and does one of the following:

- Returns a solution $\hat{\mathbf{x}} \in \mathcal{P}$ satisfying
  - $\mathbf{p}^\top \mathbf{A}\hat{\mathbf{x}} \ge \mathbf{p}^\top \mathbf{b} - \beta/3$, and
  - $\mathbf{A}_i\hat{\mathbf{x}} - b_i \in [-1, \phi]$ for *every* constraint i.
- Correctly reports that the inequality $\mathbf{p}^\top \mathbf{A}\mathbf{x} \ge \mathbf{p}^\top \mathbf{b}$ has no solution in $\mathcal{P}$.

The MWU algorithm for solving covering LPs involves T rounds. It maintains the (non-negative) weight of each constraint in $\mathbf{A}\mathbf{x} \ge \mathbf{b}$, which measures how much it has been satisfied by the solutions chosen so far. Let $\mathbf{w}^t$ denote the weight vector at the beginning of round t, and initialize the weights to $\mathbf{w}^1 \triangleq \mathbf{1}$. Then, for rounds $t = 1, \dots, T$, define the probability vector $\mathbf{p}^t$ proportional to those weights $\mathbf{w}^t$, and use the oracle above to find a solution $\mathbf{x}^t$. If the oracle reports that the system $\mathbf{p}^\top \mathbf{A}\mathbf{x} \ge \mathbf{p}^\top \mathbf{b}$ is infeasible, the MWU algorithm also reports that the original system $\mathbf{A}\mathbf{x} \ge \mathbf{b}$ is infeasible, and terminates. Otherwise, define the cost vector incurred by $\mathbf{x}^t$ as $\mathbf{m}^t \triangleq \frac{1}{\phi}(\mathbf{A}\mathbf{x} - \mathbf{b})$, then update the weights so that $w_i^{t+1} \triangleq w_i^t(1 - \beta m_i^t/6)$ and proceed to the next round. Finally, the algorithm returns the average solution $\bar{\mathbf{x}} = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}^t$.

Feasibility-SC-LP $\langle\!\langle$*Input:* $\mathcal{U},\mathcal{F},\ell\rangle\!\rangle$

$$\begin{aligned} &\sum_{S\in\mathcal{F}} x_S \le \ell && \\ &\sum_{S:e\in S} x_S \ge 1 && \forall e\in\mathcal{U} \\ &x_S \ge 0 && \forall S\in\mathcal{F} \end{aligned}$$

(a) LP relaxation of Feasibility Set Cover.

Feasibility-Covering-LP $\langle\!\langle$*Input:* $\mathbf{A},\mathbf{b},\mathbf{c},\ell\rangle\!\rangle$

$$\begin{aligned} \mathbf{c}^\top\mathbf{x} &\le \ell && \text{(objective value)} \\ \mathbf{A}\mathbf{x} &\ge \mathbf{b} && \text{(covering)} \\ \mathbf{x} &\ge \mathbf{0} && \text{(non-negativity)} \end{aligned}$$

(b) LP relaxation of the Feasibility Covering problem.

Figure 3 LP relaxations of the feasibility variant of set cover and general covering problems.

The MWU theorem (e.g., Theorem 3.5 of [6]) shows that $T = O(\frac{\phi\log n}{\beta^2})$ is sufficient to correctly solve the problem, yielding $\mathbf{A}_i\hat{\mathbf{x}} - b_i \ge -\beta$ for every constraint, where n is the number of constraints. In particular, the algorithm requires T calls to the oracle.

▶ **Theorem 2.2** (MWU Theorem [6]). *For every* $0 < \beta < 1, \phi \ge 1$ *the* MWU *algorithm either solves the* $Feasibility - Covering - LP$ *problem up to an additive error of* β *(i.e., solves* $\mathbf{A}_i\mathbf{x} - b_i \ge -\beta$ *for every* i*) or correctly reports that the LP is infeasible, making only* $O(\frac{\phi\log n}{\beta^2})$ *calls to a* $(1,\phi)$*-bounded* $\beta/3$*-approximate oracle of the LP.*

2.2 Semi Streaming MWU-based algorithm for factional Set Cover

Setting up our MWU algorithm. As described in the overview, we wish to solve, as a subroutine, the decision variant of $SetCover - LP$ known as $Feasibility - SC - LP$ given in Figure 3a, where the parameter ℓ serves as the guess for the optimal objective value.

To follow the conventional notation for solving LPs in the MWU framework, consider the more standard form of covering LPs denoted as Feasibility-Covering-LP given in Figure 3b. For our purpose, $\mathbf{A}_{n\times m}$ is the element-set incidence matrix indexed by $\mathcal{U}\times\mathcal{F}$; that is, $A_{e,S} = 1$ if $e \in S$, and $A_{e,S} = 0$ otherwise. The vectors $\mathbf{b}$ and $\mathbf{c}$ are both all-ones vectors indexed by $\mathcal{U}$ and $\mathcal{F}$, respectively. We emphasize that, unconventionally for our system $\mathbf{A}\mathbf{x} \ge \mathbf{b}$, there are n constraints (i.e. elements) and m variables (i.e. sets).

Employing the MWU approach for solving covering LPs, we define the polytope

$$\mathcal{P}_\ell \triangleq \{\mathbf{x}\in\mathbb{R}^m : \mathbf{c}^\top\mathbf{x} \le \ell \text{ and } \mathbf{x} \ge \mathbf{0}\}.$$

Observe that by applying the MWU algorithm to this polytope $\mathcal{P}$ and constraints $\mathbf{A}\mathbf{x} \ge \mathbf{b}$, we obtain a solution $\bar{\mathbf{x}} \in \mathcal{P}_\ell$ such that $\mathbf{A}_e\left(\frac{\bar{\mathbf{x}}}{1-\beta}\right) \ge \frac{b_e-\beta}{1-\beta} = 1 = b_e$, where $\mathbf{A}_e$ denotes the row of $\mathbf{A}$ corresponding to e. This yields a $(1+O(\varepsilon))$-approximate solution for $\beta = O(\varepsilon)$.

Unfortunately, we cannot implement the MWU algorithm on the full input under our streaming context. Therefore, the main challenge is to implement the following two subtasks of the MWU algorithm in the streaming settings. First, we need to design an oracle that solves the average constraint in the streaming setting. Moreover, we need to be able to efficiently update the weights for the subsequent rounds.

Covering the common elements. Before we proceed to applying the MWU framework, we add a simple first step to our implementation of **feasibilityTest** (Figure 4) that will greatly reduce the amount of sapce required in implementing the MWU algorithm. This can be interpreted as the fractional version of Set Sampling described in [17]. In our subroutine, we partition the elements into the common elements that occur more frequently, which will be

```
feasibilityTest($\ell, \varepsilon$):
  $\alpha, \beta \leftarrow \frac{\varepsilon}{3}$,   $\mathbf{p}^{\mathsf{curr}} \leftarrow \mathbf{1}_{m\times 1}$   ▷ The initial prob. vector for the MWU algorithm on $\mathcal{U}$
  ▷ Compute a cover of common elements in one pass
  $\mathbf{x}^{\mathsf{cmn}} \leftarrow \frac{\alpha\ell}{m} \cdot \mathbf{1}_{m\times 1}$,   $\mathbf{freq} \leftarrow \mathbf{0}_{n\times 1}$
  for each set $S$ in the stream do
      for each element $e \in S$ do
          $\mathsf{freq}_e \leftarrow \mathsf{freq}_e + 1$
          if $e$ appears in more than $\frac{m}{\alpha\ell}$ sets (i.e. $\mathsf{freq}_e > \frac{m}{\alpha\ell}$) then   ▷ Common element
              $p_e^{\mathsf{curr}} \leftarrow 0$
  $\mathbf{p}^{\mathsf{curr}} \leftarrow \frac{\mathbf{p}^{\mathsf{curr}}}{\|\mathbf{p}^{\mathsf{curr}}\|}$   ▷ $\mathbf{p}^{\mathsf{curr}}$ represents the current prob. vector
  $\mathbf{x}^{\mathsf{total}} \leftarrow \mathbf{0}_{m\times 1}$
  ▷ MWU algorithm for covering rare elements
  repeat $T$ times
      ▷ Solve the corresp. oracle of MWU and decide if the solution is feasible
      try $\mathbf{x} \leftarrow$ oracle($\mathbf{p}^{\mathsf{curr}}, \ell, \mathcal{F}$)
      $\mathbf{x}^{\mathsf{total}} \leftarrow \mathbf{x}^{\mathsf{total}} + \mathbf{x}$
      ▷ In one pass, update p according to x
      $\mathbf{z} \leftarrow \mathbf{0}_{n\times 1}$
      for each set $S$ in the stream do
          for each element $e \in S$ do
              $z_e \leftarrow z_e + x_S$
      if $(\mathbf{p}^{\mathsf{curr}})^\top \mathbf{z} < 1 - \beta/3$ then   ▷ Detect infeasible solutions returned by oracle
          report INFEASIBLE
      $\mathbf{p}^{\mathsf{curr}} \leftarrow$ updateProb($\mathbf{p}^{\mathsf{curr}}, \mathbf{z}$)
  $\mathbf{x}^{\mathsf{rare}} \leftarrow \frac{\mathbf{x}^{\mathsf{total}}}{(1-\beta)T}$   ▷ Scaled up the solution to cover rare elements
  return $\mathbf{x}^{\mathsf{cmn}} + \mathbf{x}^{\mathsf{rare}}$
```

Figure 4 A generic implementation of **feasibilityTest**. Its performance depend on the implementations of **oracle**, **updateProb**. We will investigate different implementations of **oracle** in the gray box.

covered if we simply choose a uniform vector solution, and the rare elements that occur less frequently, for which we perform the MWU algorithm to compute a good solution. In one pass we can find all frequently occurring elements by counting the number of sets containing each element. The amount of required space to perform this task is $O(n \log m)$.

We call an element that appears in at least $\frac{m}{\alpha\ell}$ sets *common*, and we call it *rare* otherwise, where $\alpha = \Theta(\varepsilon)$. Since we are aiming for a $(1+\varepsilon)$-approximation, we can define $\mathbf{x}^{\mathsf{cmn}}$ as a vector whose all entries are $\frac{\alpha\ell}{m}$. The total cost of $\mathbf{x}^{\mathsf{cmn}}$ is $\alpha\ell$ and all common elements are covered by $\mathbf{x}^{\mathsf{cmn}}$. Thus, throughout the algorithm we may restrict our attention to the rare elements.

Our goal now is to construct an efficient MWU-based algorithm, which finds a solution $\mathbf{x}^{\mathsf{rare}}$ covering the rare elements, with objective value at most $\frac{\ell}{1-\beta} \leq (1+\varepsilon-\alpha)\ell$. We note that our implementation does not explicitly maintain the weight vector $\mathbf{w}^t$ described in Section 2.1, but instead updates (and normalizes) its probability vector $\mathbf{p}^t$ in every round.

```
heavySetOracle($\mathbf{p}, \ell, \mathcal{F}$):
    Compute $p_S$ for every $S \in \mathcal{F}$ while reading the set system    ▷ either from stream or memory
    $S^* \leftarrow \mathbf{argmax}_{S \in \mathcal{F}} p_S$
    if $p_S < (1 - \beta/3)/\ell$ then report INFEASIBLE
    $\mathbf{x} \leftarrow \mathbf{0}_{n \times 1}, x_S \leftarrow \ell$
    return $\mathbf{x}$
```

Figure 5 **heavySetOracle** computes p_S of every set given the set system in a stream or stored memory, then returns the solution $\mathbf{x}$ that optimally places value ℓ on the corresponding entry. It reports INFEASIBLE if there is no sufficiently good solution, concluding that the set system is infeasible.

2.3 First Attempt: Simple Oracle and Large Width

A greedy solution for the oracle. We implement the oracle for MWU algorithm such that $\phi = \ell$, and thus requiring $\Theta(\ell \log n/\beta^2)$ iterations (Theorem 2.2). In each iteration, we need an oracle that finds some solution $\mathbf{x} \in \mathcal{P}_\ell$ satisfying $\mathbf{p}^\top \mathbf{A}\mathbf{x} \geq \mathbf{p}^\top \mathbf{b} - \beta/3$, or decides that no solution in $\mathcal{P}_\ell$ satisfies $\mathbf{p}^\top \mathbf{A}\mathbf{x} \geq \mathbf{p}^\top \mathbf{b}$.

Observe that $\mathbf{p}^\top \mathbf{A}\mathbf{x}$ is maximized when we place value ℓ on x_{S^*} where S^* achieves the maximum value $p_S \triangleq \sum_{e \in S} p_e$. Further, for our application, $\mathbf{b} = \mathbf{1}$ so $\mathbf{p}^\top \mathbf{b} = 1$. Our implementation **heavySetOracle** of **oracle** given in Figure 5 below is a deterministic greedy algorithm that finds a solution based on this observation. As $\mathbf{A}_e \mathbf{x} \leq \|\mathbf{x}\|_1 \leq \ell$, **heavySetOracle** implements a $(1, \ell)$-bounded $(\beta/3)$-approximate oracle. Therefore, the implementation of **feasibilityTest** with **heavySetOracle** computes a solution of objective value at most $(\alpha + \frac{1}{1-\beta})\ell < (1 + \frac{\varepsilon}{3})\ell$ when $\ell \geq k$ as promised.

Finally, we track the space usage which concludes the complexities of the current version of our algorithm: it only stores vectors of length m or n, whose entries each requires a logarithmic number of bits, yielding the following theorem.

▶ **Theorem 2.3.** *There exists a streaming algorithm that w.h.p. returns a $(1+\varepsilon)$-approximate fractional solution of SetCover $-$ LP$(\mathcal{U}, \mathcal{F})$ in $O(\frac{k \log n}{\varepsilon^2})$ passes and using $\widetilde{O}(m+n)$ memory for any positive $\varepsilon \leq 1/2$. The algorithm works in both set arrival and edge arrival streams.*

The presented algorithm suffers from large number of passes over the input. In particular, we are interested in solving the fractional Set Cover in constant number of passes using sublinear space. To this end, we first reduce the required number of rounds in MWU by a more complicated implementation of **oracle**.

3 Max Cover Problem and its Application to Width Reduction

In this section, we improve the described algorithm in the previous section and prove the following result.

▶ **Theorem 3.1.** *There exists a streaming algorithm that w.h.p. returns a $(1+\varepsilon)$-approximate fractional solution of SetCover $-$ LP$(\mathcal{U}, \mathcal{F})$ in p passes and uses $\widetilde{O}(mn^{O(1/p\varepsilon)} + n)$ memory for any $2 \leq p \leq \mathrm{polylog}(n)$ and $0 < \varepsilon \leq 1/2$. The algorithm works in both set arrival and edge arrival streams.*

Recall that in implementing **oracle**, we must find a solution $\mathbf{x}$ of total size $\|\mathbf{x}\|_1 \leq \ell$ with a sufficiently large weight $\mathbf{p}^\top \mathbf{A}\mathbf{x}$. Our previous implementation chooses only one good entry x_S and places its entire *budget* ℓ on this entry. As the width of the solution is roughly the maximum amount an element is over-covered by $\mathbf{x}$, this implementation induces a width of ℓ. In this section, we design an oracle that returns a solution in which the budget is distributed more evenly among the entries of $\mathbf{x}$ to reduce the width. To this end, we design an implementation of **oracle** of the MWU approach based on the Max ℓ-Cover problem (whose precise definition will be given shortly). The solution to our Max ℓ-Cover aids in reducing the width of our **oracle** solution to a constant, so the required number of rounds of the MWU algorithm decreases to $O(\frac{\log n}{\varepsilon^2})$, independent of ℓ. Note that, if the objective value of an optimal solution of SetCover$(\mathcal{U}, \mathcal{F})$ is ℓ, then a solution of width $o(\ell)$ may not exist, as shown in Lemma 3.2 (whose proof is given in Section A.1). This observation implies that we need to work with a different set system. Besides having small width, an optimal solution of the SetCover instance on the new set system should have the same objective value of the optimal solution of SetCover$(\mathcal{U}, \mathcal{F})$.

▶ **Lemma 3.2.** *There exists a set system in which, under the direct application of the MWU framework in computing a $(1+\varepsilon)$-approximate solution, induces width $\phi = \Omega(k)$, where k is the optimal objective value. Moreover, the exists a set system in which the approach from the previous section (which handles the frequent and rare elements differently) has width $\phi = \Theta(n) = \Theta(\sqrt{m/\varepsilon})$.*

Extended Set System. First, we consider the *extended set system* $(\mathcal{U}, \breve{\mathcal{F}})$, where $\breve{\mathcal{F}}$ is the collection containing all subsets of sets in $\mathcal{F}$; that is,

$$\breve{\mathcal{F}} \triangleq \{R : R \subseteq S \text{ for some } S \in \mathcal{F}\}.$$

It is straightforward to see that the optimal objective value of SetCover over $(\mathcal{U}, \breve{\mathcal{F}})$ is equal to that of $(\mathcal{U}, \mathcal{F})$: we only add subsets of the original sets to create $\breve{\mathcal{F}}$, and we may replace any subset from $\breve{\mathcal{F}}$ in our solution with its original set in $\mathcal{F}$. Moreover, we may *prune* any collection of sets from $\mathcal{F}$ into a collection from $\breve{\mathcal{F}}$ of the same cardinality so that, this pruned collection not only covers the same elements, but also each of these elements is covered exactly once. This extended set system is defined for the sake of analysis only: we will never explicitly handle an exponential number of sets throughout our algorithm.

We define *ℓ-cover* as a collection of sets of total weight ℓ. Although the pruning of an ℓ-cover reduces the width, the total weight $\mathbf{p}^\top \mathbf{A}\mathbf{x}$ of the solution will decrease. Thus, we consider the weighted constraint of the form

$$\sum_{e \in \mathcal{U}} \left(p_e \cdot \min\{1, \sum_{S : e \in S} x_S\} \right) \geq 1;$$

that is, we can only gain the value p_e without any multiplicity larger than 1. The problem of maximizing the left hand side is known as the *weighted max coverage* problem: for a parameter ℓ, find an ℓ-cover such that the total value p_e's of the covered elements is maximized.

3.1 The Maximum Coverage Problem

In the design of our algorithm, we consider the *weighted* Max k-Cover problem, which is closely related to SetCover. Extending upon the brief description given earlier, we fully specify the LP relaxation of this problem. In the weighted Max k-Cover$(\mathcal{U}, \mathcal{F}, \ell, \mathbf{p})$, given a ground set of elements $\mathcal{U}$, a collection of sets $\mathcal{F}$ over the ground set, a budget parameter ℓ, and a weight vector $\mathbf{p}$, the goal is to return ℓ sets in $\mathcal{F}$ whose weighted *coverage*, the total weight of all covered elements, is maximized. Moreover, since we are aiming for a

$$\begin{array}{lll}
\multicolumn{3}{l}{\text{MaxCover-LP} \quad \langle\!\langle \textit{Input: } \mathcal{U}, \mathcal{F}, \ell, \mathbf{p} \rangle\!\rangle} \\
\text{maximize} & \displaystyle\sum_{e \in \mathcal{U}} p_e z_e & \\
\text{subject to} & \displaystyle\sum_{S : e \in S} x_S \geq z_e & \forall e \in \mathcal{U} \\
 & \displaystyle\sum_{S \in \mathcal{F}} x_S = \ell & \\
 & 0 \leq z_e \leq 1 & \forall e \in \mathcal{U} \\
 & x_S \geq 0 & \forall S \in \mathcal{F}
\end{array}$$

Figure 6 LP relaxation of weighted Max k-Cover.

fractional solution of Set Cover, we consider the LP relaxation of weighted Max k-Cover, $MaxCover - LP$ (see Figure 6); in this LP relaxation, z_e denotes the fractional amount that an element is covered, and hence is capped at 1.

As an intermediate goal, we aim to compute an approximate solution of $MaxCover - LP$, given that the optimal solution covers all elements in the ground set, or to correctly detect that no solution has weighted coverage of more than $(1 - \varepsilon)$. In our application, the vector $\mathbf{p}$ is always a probability vector: $\mathbf{p} \geq \mathbf{0}$ and $\sum_{e \in \mathcal{U}} p_e = 1$. We make the following useful observation.

▶ **Observation 3.3.** *Let k be the value of an optimal solution of $SetCover - LP(\mathcal{U}, \mathcal{F})$ and let $\mathbf{p}$ be an arbitrary probability vector over the ground set. Then there exists a fractional solution of $MaxCover - LP(\mathcal{U}, \mathcal{F}, \ell, \mathbf{p})$ whose weighted coverage is one if $\ell \geq k$.*

δ-integral near optimal solution of MaxCover-LP. Our plan is to solve MaxCover-LP over a randomly projected set system, and argue that with high probability this will result in a valid **oracle**. Such an argument requires an application of the union bound over the set of solutions, which is generally of unbounded size. To this end, we consider a more restrictive domain of δ-*integral* solutions: this domain has bounded size, but is still guaranteed to contain a sufficiently good solution.

▶ **Definition 3.4** (δ-integral solution)**.** A fractional solution $\mathbf{x}_{n \times 1}$ of an LP is δ-integral if $\frac{1}{\delta} \cdot \mathbf{x}$ is an integral vector. That is, for each $i \in [n]$, $x_i = v_i \delta$ where each v_i is an integer.

Next we claim that **maxCoverOracle** given in Figure 7 below, which is the MWU algorithm with **heavySetOracle** for solving $MaxCover - LP$, results in a δ-integral solution. The proof of the following lemma is given in Section A.2.

▶ **Lemma 3.5.** *Consider a $MaxCover - LP$ with the optimal objective value* OPT *(where the weights of elements form a probability vector). There exists a $\Theta(\frac{\varepsilon_{\mathrm{MC}}^2}{\log n})$-integral solution of $MaxCover - LP$ whose objective value is at least $(1 - \varepsilon_{\mathrm{MC}})$OPT. In particular, if an optimal solution covers all elements $\mathcal{U}$ ($\ell \geq k$),* **maxCoverOracle** *returns a solution whose weighted coverage is at least $1 - \varepsilon_{\mathrm{MC}}$ in polynomial time.*

Pruning a fractional ℓ-cover. In our analysis, we aim to solve the Set Cover problem under the extended set system. We claim that any solution $\mathbf{x}$ with coverage $\mathbf{z}$ in the actual set system may be turned into a pruned solution $\check{\mathbf{x}}$ in the extended set system that provides

```
**maxCoverOracle**($\mathcal{U}, \mathcal{F}, \ell$):
  $\mathbf{x} \leftarrow$ MWU solution of Set Cover LP relaxation implemented with **heavySetOracle**
  **return** $\mathbf{x}$
```

Figure 7 **maxCoverOracle** returns a fractional ℓ-cover with weighted coverage at least $1-\beta/3$ w.h.p. if $\ell \geq k$. It provides no guarantee on its behavior if $\ell < k$.

the same coverage $\mathbf{z}$, but satisfies the strict equality $\sum_{\breve{S}\in\breve{\mathcal{F}}:e\in\breve{S}} \breve{x}_{\breve{S}} = z_e$. Since $z_e \leq 1$, the pruned solution satisfies the condition for an oracle with width *one*. We give an algorithm **prune** for pruning $\mathbf{x}$ into $\breve{\mathbf{x}}$ in Section A.3 and only state the property of this algorithm here.

▶ **Lemma 3.6.** *A fractional ℓ-cover $\mathbf{x}$ of $(\mathcal{U}, \mathcal{F})$ can be converted, in polynomial time, to a fractional ℓ-cover $\breve{\mathbf{x}}$ of $(\mathcal{U}, \breve{\mathcal{F}})$ such that for each element e, its coverage $z_e = \sum_{\breve{S}\in\breve{\mathcal{F}}:e\in\breve{S}} \breve{x}_{\breve{S}} = \min(\sum_{S:e\in S} x_S, 1)$.*

We remark that in order to update the weights in the MWU framework, it is sufficient to know the vector $\mathbf{z}$, which has a simple formula given in the lemma above. The actual solution $\breve{\mathbf{x}}$ is not necessary.

3.2 Sampling-Based Oracle for Fractional Max Coverage

In the previous section, we simply needed to compute the values p_S's in order to construct a solution for the **oracle**. Here as we aim to bound the width of **oracle**, our new task is to find a fractional ℓ-cover $\mathbf{x}$ whose weighted coverage is at least $1-\beta/3$. The *element sampling* technique, which is also known from prior work in streaming Set Cover and Max k-Cover, is to sample a few elements and solve the problem over the sampled elements only. Then, by applying the union bound over all possible candidate solutions, it is shown that w.h.p. a nearly optimal cover of the sampled elements also covers a large fraction of the whole ground set. This argument applies to the aforementioned problems precisely because there are standard ways of bounding the number of all integral candidate solutions (e.g. ℓ-covers).

However, in the fractional setting, there are infinitely many solutions. Consequently, we employ the notion of δ-integral solutions where the number of such solutions is bounded. In Lemma 3.6, we showed that there always exists a δ-integral solution to $MaxCover-LP$ whose coverage is at least a $(1-\varepsilon_{\mathrm{MC}})$-fraction of an optimal solution. Moreover, the number of all possible solutions is bounded by the number of ways to divide the budget ℓ into ℓ/δ equal parts of value δ and distribute them (possibly with repetition) among m entries:

▶ **Observation 3.7.** *The number of feasible δ-integral solutions to $MaxCover-LP(\mathcal{U}, \mathcal{F}, \ell, \mathbf{p})$ is $O(m^{\ell/\delta})$ for any multiple ℓ of δ.*

Next, we design our algorithm using the element sampling technique: we show that a $(1-\beta/3)$-approximate solution of $MaxCover-LP$ can be computed using the projection of all sets in $\mathcal{F}$ over a set of elements of size $\Theta(\frac{\ell \log n \log mn}{\beta^4})$ picked according to $\mathbf{p}$. For every fractional solution $(\mathbf{x}, \mathbf{z})$ and subset of elements $\mathcal{V} \subseteq \mathcal{U}$, let $\mathcal{C}_{\mathcal{V}}(\mathbf{x}) \triangleq \sum_{e\in\mathcal{V}} p_e z_e$ denote the coverage of elements in $\mathcal{V}$ where $z_e = \min(1, \sum_{S:e\in S} x_S)$. We may omit the subscript $\mathcal{V}$ in $\mathcal{C}_{\mathcal{V}}$ if $\mathcal{V} = \mathcal{U}$.

The following lemma, which is essentially an extension of the Element Sampling lemma of [17] for our application, MaxCover-LP, shows that a $(1-\varepsilon_{\mathrm{MC}})$-approximate ℓ-cover over a set of sampled elements of size $\Theta(\ell \log n \log mn/\gamma^4)$ w.h.p. has a weighted coverage of at least $(1-2\gamma)(1-\varepsilon_{\mathrm{MC}})$ if there exists a fractional ℓ-cover whose coverage is 1. Thus, choosing $\varepsilon_{\text{MC}} = \gamma = \beta/9$ yields the desired guarantee for **maxCoverOracle**, leading to the performance given in Theorem 3.9. The omitted proofs are given in Section A.4-A.5.

▶ **Lemma 3.8.** *Let ε_{MC} and γ be parameters. Consider the $MaxCover - LP(\mathcal{U}, \mathcal{F}, \ell, \mathbf{p})$ with optimal solution of value* OPT*, and let $\mathcal{L}$ be a multi-set of $s = \Theta(\ell \log n \log(mn)/\gamma^4)$ elements sampled independently at random according to the probability vector $\mathbf{p}$. Let $\mathbf{x}^{\text{SOL}}$ be a $(1 - \varepsilon_{\text{MC}})$-approximate $\Theta(\frac{\gamma^2}{\log n})$-integral ℓ-cover over the sampled elements. Then with high probability, $\mathcal{C}(\mathbf{x}^{\text{SOL}}) \geq (1 - 2\gamma)(1 - \varepsilon_{\text{MC}})\text{OPT}$.*

▶ **Theorem 3.9.** *There exists a streaming algorithm that w.h.p. returns a $(1+\varepsilon)$-approximate fractional solution of $SetCover - LP(\mathcal{U}, \mathcal{F})$ in $O(\log n/\varepsilon^2)$ passes and uses $\widetilde{O}(m/\varepsilon^6 + n)$ memory for any positive $\varepsilon \leq 1/2$. The algorithm works in both set arrival and edge arrival streams.*

3.3 Final Step: Running Several MWU Rounds Together

We complete our result by further reducing the number of passes at the expense of increasing the required amount of memory, yielding our full algorithm **fastFeasibilityTest** in Figure 8. More precisely, aiming for a p-pass algorithm, we show how to execute $R \triangleq \frac{T}{\Theta(p)} = \Theta(\frac{\log n}{p\beta^2})$ rounds of the MWU algorithm in a single pass. We show that this task may be accomplished with a multiplicative factor of $f \cdot \Theta(\log mn)$ increase in memory usage, where $f \triangleq n^{\Theta(1/(p\beta))}$.

Advance sampling. Consider a sequence of R consecutive rounds $i = 1, \ldots, R$. In order to implement the MWU algorithm for these rounds, we need (multi-)sets of sampled elements $\mathcal{L}_1, \ldots, \mathcal{L}_R$ according to probabilities $\mathbf{p}^1, \ldots, \mathbf{p}^R$, respectively (where $\mathbf{p}^i$ is the probability corresponding to round i). Since the probabilities of subsequent rounds are not known in advance, we circumvent this problem by choosing these sets $\mathcal{L}_i$'s with probabilities according to $\mathbf{p}^1$, but the number of samples in each set will be $|\mathcal{L}_i| = s \cdot f \cdot \Theta(\log mn)$ instead of s. Then, once $\mathbf{p}^i$ is revealed, we sub-sample the elements from $\mathcal{L}_i$ to obtain $\mathcal{L}'_i$ as follow: for a (copy of) sampled element $\hat{e} \in \mathcal{L}_i$, add $\hat{e}$ to $\mathcal{L}'_i$ with probability $\frac{p^i_{\hat{e}}}{p^1_{\hat{e}} f}$; otherwise, simply discard it. Note that it is still left to be shown that the probability above is indeed at most 1.

Since each e was originally sampled with probability p^1_e, then in $\mathcal{L}'_i$, the probability that a sampled element $\hat{e} = e$ is exactly p^i_e/f. By having $f \cdot \Theta(\log mn)$ times the originally required number of samples s in the first place, in expectation we still have $\mathbf{E}[|\mathcal{L}'_i|] = |\mathcal{L}_i| \sum_{e \in \mathcal{U}} \frac{p^i_e}{f} = (s \cdot f \cdot \Theta(\log mn)) \frac{1}{f} = s \cdot \Theta(\log mn)$. Due to the $\Theta(\log mn)$ factor, by the Chernoff bound, we conclude that with w.h.p. $|\mathcal{L}'_i| \geq s$. Thus, we have a sufficient number of elements sampled with probability according to $\mathbf{p}^i$ to apply Lemma 3.8, as needed.

Change in probabilities. As noted above, we must show that the probability that we sub-sample each element is at most 1; that is, $p^i_e/p^1_e \leq f = n^{\Theta(1/(p\beta))}$ for every element e and every round $i = 1, \ldots, R$. We bound the multiplicative difference between the probabilities of two consecutive rounds as follows (see Section A.6 for proof).

▶ **Lemma 3.10.** *Let $\mathbf{p}$ and $\mathbf{p}'$ be the probability of elements before and after an update. Then for every element e, $p'_e \leq (1 + O(\beta))p_e$.*

Therefore, after $R = \Theta(\frac{\log n}{p\beta^2})$ rounds, the probability of any element may increase by at most a factor of $(1 + O(\beta))^{\Theta(\frac{\log n}{p\beta^2})} \leq e^{\Theta(\frac{\log n}{p\beta})} = n^{\Theta(1/(p\beta))} = f$, as desired. This concludes the proof of Theorem 3.1.

Implementation details. We make a few remarks about the implementation given in Figure 8. First, even though we perform all sampling in advance, the decisions of **maxCoverOracle** do not depend on any $\mathcal{L}_i$ of later rounds, and **updateProb** is entirely deterministic: there is no dependency issue between rounds. Next, we only need to perform **updateProb** on the sampled elements $\mathcal{L} = \mathcal{L}_1 \cup \cdots \cup \mathcal{L}_R$ during the current R rounds. We therefore denote the probabilities with a different vector $\mathbf{q}^i$ over the sampled elements $\mathcal{L}$ only. Probabilities of elements outside $\mathcal{L}$ are not required by **maxCoverOracle** during these rounds, but we simply need to spend one more pass after executing R rounds of MWU to aggregate the new probability vector $\mathbf{p}$ over all (rare) elements. Similarly, since **maxCoverOracle** does not have the ability to verify, during the MWU algorithm, that each solution $\mathbf{x}^i$ returned by the oracle indeed provides a sufficient coverage, we check all of them during this additional pass. Lastly, we again remark that this algorithm operates on the extended set system: the solution $\mathbf{x}$ returned by **maxCoverOracle** has at least the same coverage as $\check{\mathbf{x}}$. While $\check{\mathbf{x}}$ is not explicitly computed, its coverage vector $\mathbf{z}$ can be computed exactly.

3.4 Extension to general covering LPs

We remark that our MWU-based algorithm can be extended to solve a more general class of covering LPs. Consider the problem of finding a vector $\mathbf{x}$ that minimizes $\mathbf{c}^\top \mathbf{x}$ subject to constraints $\mathbf{A}\mathbf{x} \geq \mathbf{b}$ and $\mathbf{x} \geq \mathbf{0}$. In terms of the Set Cover problem, $A_{e,S} \geq 0$ indicates the multiplicity of an element e in the set S, $b_e > 0$ denotes the number of times we wish e to be covered, and $c_S > 0$ denotes the cost per unit for the set S. Now define

$$L \triangleq \min_{(e,S): A_{e,S} \neq 0} \frac{A_{e,S}}{b_e c_S} \quad \text{and} \quad U \triangleq \max_{(e,S)} \frac{A_{e,S}}{b_e c_S}.$$

Then, we may modify our algorithm to obtain the following result.

▶ **Theorem 3.11.** *There exists a streaming algorithm that w.h.p. returns a $(1+\varepsilon)$-approximate fractional solution to general covering LPs in p passes and using $\widetilde{O}(\frac{mU}{\varepsilon^6 L} \cdot n^{O(\frac{1}{p\varepsilon})} + n)$ memory for any $3 \leq p \leq \mathrm{polylog}(n)$, where parameters L and U are defined above. The algorithm works in both set arrival and edge arrival streams.*

Proof. We modify our algorithm and provide an argument of its correctness as follows. First, observe that we can convert the input LP into an equivalent LP with all entries $b_e = c_S = 1$ by simply replacing each $A_{e,S}$ with $\frac{A_{e,S}}{b_e c_S}$. Namely, let the new parameters be $\mathbf{A}', \mathbf{b}'$ and $\mathbf{c}'$, and we consider the variable $\mathbf{x}'$ where $x'_S = c_S x_S$. It is straightforward to verify that $\mathbf{c}'^\top \mathbf{x}' = \mathbf{c}^\top \mathbf{x}$ and $\mathbf{A}'_e \mathbf{x}' = \frac{\mathbf{A}_e \mathbf{x}}{b_e}$, reducing the LP into the desired case. Thus, we may afford to record $\mathbf{b}$ and $\mathbf{c}$, so that each value $\frac{A_{e,S}}{b_e c_S}$ may be computed on-the-fly. Henceforth we assume that all entries $b_e = c_S = 1$ and $A_{e,S} \in \{0\} \cup [L, U]$. Observe as well that the optimal objective value k may be in the expanded range $[1/U, n/L]$, so the number of guesses must be increased from $\frac{\log n}{\varepsilon}$ to $\frac{\log(nU/L)}{\varepsilon}$.

Next consider the process for covering the rare elements. We instead use a uniform solution $\mathbf{x}^{\mathsf{cmn}} = \frac{\alpha \ell L}{m} \cdot \mathbf{1}$. Observe that if an element occurs in at least $\frac{m}{\alpha \ell L}$ sets, then $\mathbf{A}_e \mathbf{x}^{\mathsf{cmn}} = \sum_{S: e \in S} A_{e,S} \cdot \frac{\alpha \ell}{m} \geq \frac{m}{\alpha \ell L} \cdot L \cdot \frac{\alpha \ell}{m} = 1$. That is, we must adjust our definition so that an element is considered common if it appears in at least $\frac{m}{\alpha \ell L}$ sets. Consequently, whenever we perform element sampling, the required amount of memory to store information of each element increases by a factor of $1/L$.

Next consider Lemma 3.5, where we show an existence of integral solutions via the MWU algorithm with a greedy oracle. As the greedy implementation chooses a set S and places the entire budget ℓ on x_S, the amount of coverage $A_{e,S} x_S$ may be as large as ℓU as $A_{e,S}$ is

```
fastFeasibilityTest($\ell, \varepsilon$):
  $\alpha, \beta \leftarrow \frac{\varepsilon}{3}$,   $\mathbf{p}^{\mathsf{curr}} \leftarrow \mathbf{1}_{m\times 1}$  ▷ The initial prob. vector for the MWU algorithm on $\mathcal{U}$
  Compute a cover of common elements in one pass  ▷ See Fig. 4's feasibilityTest block
  $\mathbf{x}^{\mathsf{total}} \leftarrow \mathbf{0}_{m\times 1}$
  ▷ MWU algorithm for covering rare elements
  repeat $p$ times
    $R \leftarrow \Theta(\frac{\log n}{p\beta^2})$  ▷ Number of MWU iterations performed together
    ▷ In one pass, projects all sets in $\mathcal{F}$ over the collections of samples $\mathcal{L}_1, \cdots \mathcal{L}_R$
    sample $\mathcal{L}_1, \ldots, \mathcal{L}_R$ according to $\mathbf{p}^{\mathsf{curr}}$ each of size $\ell n^{\Theta(1/(p\beta))}$ poly$(\log mn)$
    $\mathcal{L} \leftarrow \mathcal{L}_1 \cup \cdots \cup \mathcal{L}_R$,   $\mathcal{F}_{\mathcal{L}} \leftarrow \emptyset$  ▷ $\mathcal{L}$ is a set whereas $\mathcal{L}_1, \ldots, \mathcal{L}_R$ are multi-sets
    for each set $S$ in the stream do $\mathcal{F}_{\mathcal{L}} \leftarrow \mathcal{F}_{\mathcal{L}} \cup \{S \cap \mathcal{L}\}$
    ▷ Each pass simulates $R$ rounds of MWU
    for each $e \in \mathcal{L}$ do $q_e^1 \leftarrow p_e^{\mathsf{curr}}$  ▷ Project $\mathbf{p}^{\mathsf{curr}}_{n\times 1}$ to $\mathbf{q}^1_{|\mathcal{L}|\times 1}$ over sampled elements
    $\mathbf{q}^1 \leftarrow \frac{\mathbf{q}^1}{\|\mathbf{q}^1\|}$
    for each round $i = 1, \ldots, R$ do
      $\mathcal{L}_i' \leftarrow$ sample each elt $e \in \mathcal{L}_i$ with probab. $\frac{q_e^i}{q_e^1 n^{\Theta(1/(p\beta))}}$  ▷ Rejection Sampling
      $\mathbf{x}^i \leftarrow$ maxCoverOracle$(\mathcal{L}_i', \mathcal{F}_{\mathcal{L}}, \ell)$  ▷ w.h.p. $\mathcal{C}(\mathbf{x}^i) \geq 1 - \beta/3$ when $\ell \geq k$
      ▷ In no additional pass, updates probab. $\mathbf{q}$ over sampled elts according to $\mathbf{x}^i$
      $\mathbf{z} \leftarrow \mathbf{0}_{|\mathcal{L}|\times 1}$  ▷ Compute coverage over sampled elements
      for each element-set pair $e \in S$ where $S \in \mathcal{F}_{\mathcal{L}}$ do $z_e \leftarrow \min(z_e + x_S^i, 1)$
      $\mathbf{q}^{i+1} \leftarrow$ updateProb$(\mathbf{q}^i, \mathbf{z})$  ▷ Only update weights of elements in $\mathcal{L}$
    ▷ In one pass, updates probab. $\mathbf{p}^{\mathsf{curr}}$ over all (rare) elts according to $\mathbf{x}^1, \ldots, \mathbf{x}^R$
    $\mathbf{z}^1, \ldots, \mathbf{z}^R \leftarrow \mathbf{0}_{n\times 1}$  ▷ Compute coverage over all (rare) elements
    for each element-set pair $e \in S$ in the stream do
      for each round $i = 1, \ldots, R$ do $z_e^i \leftarrow \min(z_e^i + x_S^i, 1)$
    for each round $i = 1, \ldots, R$ do
      if $(\mathbf{p}^{\mathsf{curr}})^\top \mathbf{z}^i < 1 - \beta/3$ then  ▷ Detect infeasible solutions
        report INFEASIBLE
      $\mathbf{x}^{\mathsf{total}} \leftarrow \mathbf{x}^{\mathsf{total}} + \mathbf{x}^i$, $\mathbf{p}^{\mathsf{curr}} \leftarrow$ updateProb$(\mathbf{p}^{\mathsf{curr}}, \mathbf{z}^i)$  ▷ Perform actual updates
  $\mathbf{x}^{\mathsf{rare}} \leftarrow \frac{\mathbf{x}^{\mathsf{total}}}{(1-\beta)T}$  ▷ Scaled up the solution to cover rare elements
  return $\mathbf{x}^{\mathsf{cmn}} + \mathbf{x}^{\mathsf{rare}}$
```

Figure 8 An efficient implementation of **feasibilityTest** which performs in p passes and consumes $\widetilde{O}(mn^{O(\frac{1}{p\varepsilon})} + n)$ space.

no longer bounded by 1. Thus this application of the MWU algorithm has width $\phi = \Theta(\ell U)$ and requires $T = \Theta(\frac{\ell U \log n}{\varepsilon_{\mathrm{MC}}^2})$ rounds. Consequently, its solution becomes $\Theta(\frac{\ell}{T}) = \Theta(\frac{\varepsilon_{\mathrm{MC}}^2}{U \log n})$-integral. As noted in Observation 3.7, the number of potential solutions from the greedy oracle increases by a power of U. Then, in Lemma 3.8, we must reduce the error probability of each solution by the same power. We increase the number of samples s by a factor of U to account for this change, increasing the required amount of memory by the same factor.

As in the previous case, any solution $\mathbf{x}$ may always be pruned so that the width is reduced to 1: our algorithm **prune** still works as long as the entries of $\mathbf{A}$ are non-negative (Section A.3). Therefore, the fact that entries of $\mathbf{A}$ may take on values other than 0 or 1 does not affect the number of rounds (or passes) of our overall application of the MWU framework. Thus, we may handle general covering LPs using a factor of $\widetilde{O}(U/L)$ larger memory within the same number of passes. In particular, if the non-zero entries of the input are bounded in the range $[1, M]$, this introduces a factor of $\widetilde{O}(U/L) \leq \widetilde{O}(M^3)$ overhead in memory usage. ◀

References

1 K. J. Ahn and S. Guha. Linear programming in the semi-streaming model with application to the maximum matching problem. In *Proc. 38th Int'l Colloq. Automata Lang. Prog.* (ICALP), pages 526–538. Springer, 2011.

2 K. J. Ahn and S. Guha. Access to data and number of iterations: Dual primal algorithms for maximum matching under resource constraints. In *Proc. 27th ACM Symp. Parallel Alg. Arch.* (SPAA), pages 202–211, 2015.

3 Z. Allen-Zhu and L. Orecchia. Nearly-linear time positive LP solver with faster convergence rate. In *Proc. 47th Annual ACM Symp. Theory Comput.* (STOC), pages 229–236, 2015.

4 Z. Allen-Zhu and L. Orecchia. Using optimization to break the epsilon barrier: A faster and simpler width-independent algorithm for solving positive linear programs in parallel. In *Proc. 26th ACM-SIAM Symp. Discrete Algs.* (SODA), pages 1439–1456, 2015.

5 N. Alon, D. Moshkovitz, and S. Safra. Algorithmic construction of sets for k-restrictions. *ACM Trans. Algo.*, 2(2):153–177, 2006.

6 S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

7 S. Assadi. Tight space-approximation tradeoff for the multi-pass streaming set cover problem. In *Proc. 36th ACM Symp. on Principles of Database Systems* (PODS), pages 321–335, 2017.

8 S. Assadi, S. Khanna, and Y. Li. Tight bounds for single-pass streaming complexity of the set cover problem. In *Proc. 48th Annual ACM Symp. Theory Comput.* (STOC), pages 698–711, 2016.

9 A. Badanidiyuru, B. Mirzasoleiman, A. Karbasi, and A. Krause. Streaming submodular maximization: Massive data summarization on the fly. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 671–680, 2014.

10 B. Bahmani, A. Goel, and K. Munagala. Efficient primal-dual graph algorithms for mapreduce. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 59–78. Springer, 2014.

11 N. Bansal, A. Caprara, and M. Sviridenko. A new approximation method for set covering problems, with applications to multidimensional bin packing. *SIAM Journal on Computing*, 39(4):1256–1278, 2009.

12 M. Bateni, H. Esfandiari, and V. S. Mirrokni. Almost optimal streaming algorithms for coverage problems. *Proc. 29th ACM Symp. Parallel Alg. Arch.* (SPAA), 2017.

13 A. Chakrabarti and A. Wirth. Incidence geometries and the pass complexity of semi-streaming set cover. In *Proc. 27th ACM-SIAM Symp. Discrete Algs.* (SODA), pages 1365–1373, 2016.

14 C. Chekuri, S. Gupta, and K. Quanrud. Streaming algorithms for submodular function maximization. In *Proc. 42st Int'l Colloq. Automata Lang. Prog.* (ICALP), pages 318–330. Springer, 2015.

15 F. Chierichetti, R. Kumar, and A. Tomkins. Max-cover in map-reduce. In *Proc. 19th Int. Conf. World Wide Web* (WWW), pages 231–240, 2010.

16 G. Cormode, H. J. Karloff, and A. Wirth. Set cover algorithms for very large datasets. In *Proc. 19th ACM Conf. Info. Know. Manag.* (CIKM), pages 479–488, 2010.

17 E. D. Demaine, P. Indyk, S. Mahabadi, and A. Vakilian. On streaming and communication complexity of the set cover problem. In *Proc. 28th Int'l Symp. Dist. Comp.* (DISC), volume 8784 of *Lect. Notes in Comp. Sci.*, pages 484–498, 2014.

18 I. Dinur and D. Steurer. Analytical approach to parallel repetition. In *Proc. 46th Annual ACM Symp. Theory Comput.* (STOC), pages 624–633. ACM, 2014.

19 P. Drineas, R. Kannan, and M. W. Mahoney. Sampling sub-problems of heterogeneous max-cut problems and approximation algorithms. In *Proc. 37th Annual ACM Symp. Theory Comput.* (STOC), pages 57–68. Springer, 2005.

20 Y. Emek and A. Rosén. Semi-streaming set cover. In *Proc. 41st Int'l Colloq. Automata Lang. Prog.* (ICALP), volume 8572 of *Lect. Notes in Comp. Sci.*, pages 453–464, 2014.

21 U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

22 N. Garg and J. Koenemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM Journal on Computing (SIAM)*, 37(2):630–652, 2007.

23 T. Grossman and A. Wool. Computational experience with approximation algorithms for the set covering problem. *Euro. J. Oper. Res.*, 101(1):81–92, 1997.

24 S. Har-Peled, P. Indyk, S. Mahabadi, and A. Vakilian. Towards tight bounds for the streaming set cover problem. In *Proc. 35th ACM Symp. on Principles of Database Systems* (PODS), pages 371–383, 2016.

25 N. Karmarkar and R. M. Karp. An efficient approximation scheme for the one-dimensional bin-packing problem. In *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*, pages 312–320. IEEE, 1982.

26 M. J. Kearns and U. V. Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

27 C. Koufogiannakis and N. E. Young. A nearly linear-time PTAS for explicit fractional packing and covering linear programs. *Algorithmica*, 70(4):648–674, 2014.

28 Y. T. Lee and A. Sidford. Path finding methods for linear programming: Solving linear programs in $O(vrank)$ iterations and faster algorithms for maximum flow. In *Proc. 55th Annual IEEE Symp. Found. Comput. Sci.* (FOCS), pages 424–433, 2014.

29 C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM (JACM)*, 41(5):960–981, 1994.

30 A. McGregor and H. T. Vu. Better streaming algorithms for the maximum coverage problem. In *20th International Conference on Database Theory, ICDT*, pages 22:1–22:18, 2017.

31 D. Moshkovitz. The projection games conjecture and the NP-hardness of $\ln n$-approximating set-cover. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 276–287. Springer, 2012.

32 S. A. Plotkin, D. B. Shmoys, and É. Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20(2):257–301, 1995.

33 R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proc. 29th Annual ACM Symp. Theory Comput.* (STOC), 1997.

34 B. Saha and L. Getoor. On maximum coverage in the streaming model & application to multi-topic blog-watch. In *Proc. SIAM Int. Conf. Data Mining* (SDM), pages 697–708, 2009.

35 D. Wang, S. Rao, and M. W. Mahoney. Unified acceleration method for packing and covering problems via diameter reduction. In *Proc. 43st Int'l Colloq. Automata Lang. Prog.* (ICALP), pages 50:1–50:13, 2016.

36 N. E. Young. Randomized rounding without solving the linear program. In *Proceedings of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 170–178, 1995.

37 N. E. Young. Nearly linear-work algorithms for mixed packing/covering and facility-location linear programs. *arXiv preprint arXiv:1407.3015*, 2014.

A Omitted Proofs

A.1 Proof of Lemma 3.2

Proof. For the first claim, we consider an arbitrary set system, then modify it by adding a common element e to all sets. Recall that the MWU framework returns an average of the solutions from all rounds. Thus there must exist a round where the oracle returns a solution $\mathbf{x}$ of size $\|\mathbf{x}\|_1 = \Theta(k)$. For the added element e, this solution has $\sum_{S:e\in S} x_S = \sum_{S\in\mathcal{F}} x_S = \Theta(k)$, inducing width $\phi = \Omega(k)$.

For the second claim, consider the following set system with $k = \sqrt{m/\varepsilon}$ and $n = 2k+1$. For $i = 1,\ldots,k$, let $S_i = \{e_i, e_{k+i}, e_{2k+1}\}$, whereas the remaining $m-k$ sets are arbitrary subsets of $\{e_1,\ldots,e_k\}$. Observe that e_{k+i} is contained only in S_i, so $x_{S_i} = 1$ in any valid set cover. Consequently the solution $\mathbf{x}$ where $x_{S_1} = \cdots = x_{S_k} = 1$ and $x_{S_{k+1}} = \cdots = x_{S_{2k+1}} = 0$ forms the unique (fractional) minimum set cover of size $k = \sqrt{m/\varepsilon}$. Next, recall that an element is considered rarely occurring if it appears in at most $\frac{m}{\alpha\ell} > \frac{m}{\varepsilon k}$ sets. As $e_{k+1},\ldots,e_{2k}$ each only occurs once, and e_{2k+1} only appears in $k = \sqrt{m/\varepsilon} = \frac{m}{\varepsilon k}$ sets, these $k+1$ elements are deemed rare and thus handled by the MWU framework.

The solution computed by the MWU framework satisfies $\sum_{S:e\in S} x_S \geq 1-\beta$ for every e, and in particular, for each $e \in \{e_{k+1},\ldots,e_{2k}\}$. Therefore, the average solution places a total weight of at least $(1-\beta)\cdot\Theta(k)$ on $x_{S_1},\ldots,x_{S_k}$, so there must exist a round that places at least the same total weight on these sets. However, these k sets all contain e_{2k+1}, yielding $\sum_{S:e_{2k+1}\in S} x_S \geq (1-\beta)\cdot\Theta(k) = \Omega(k)$, implying a width of $\Omega(k) = \Omega(\sqrt{m/\varepsilon})$. ◀

A.2 Proof of Lemma 3.5

Proof. Let $(\mathbf{x}^*, \mathbf{z}^*)$ denote the optimal solution of value OPT to $MaxCover - LP$, which implies that $\|\mathbf{x}^*\|_1 \leq \ell$ and $\mathbf{A}\mathbf{x}^* \geq \mathbf{z}^*$. Consider the following covering LP: minimize $\|\mathbf{x}\|_1$ subject to $\mathbf{A}\mathbf{x} \geq \mathbf{z}^*$ and $\mathbf{x} \geq \mathbf{0}$. Clearly there exists an optimal solution of objective value ℓ, namely $\mathbf{x}^*$. This covering LP may be solved via the MWU framework. In particular, we may use the oracle that picks one set S with maximum weight (as maintained in the MWU framework) and places its entire budget on x_S. For an accurate guess $\ell' = \Theta(\ell)$ of the optimal value, this algorithm returns an average of $T = \Theta(\frac{\ell' \log n}{\varepsilon_{\mathrm{MC}}^2}) = \Theta(\frac{\ell \log n}{\varepsilon_{\mathrm{MC}}^2})$ oracle solutions. Observe that the outputted solution $\mathbf{x}$ is of the form $x_S = \frac{v_S \ell'}{T} = v_S\delta$ where v_S is the number of rounds in which S is chosen by the oracle, and $\delta = \frac{\ell'}{T} = \frac{\ell' \varepsilon_{\mathrm{MC}}^2}{\ell \log n} = \Theta(\frac{\varepsilon_{\mathrm{MC}}^2}{\log n})$. In other words, $\mathbf{x}$ is $(\frac{\varepsilon_{\mathrm{MC}}^2}{\log n})$-integral. By Theorem 2.2, $\mathbf{x}$ satisfies $\mathbf{A}\mathbf{x} \geq (1-\varepsilon_{\mathrm{MC}})\mathbf{z}^*$. Then in $MaxCover - LP$, the solution $(\mathbf{x}, (1-\varepsilon_{\mathrm{MC}})\mathbf{z}^*)$ yields coverage at least $\mathbf{p}^\top((1-\varepsilon_{\mathrm{MC}})\mathbf{z}^*) = (1-\varepsilon_{\mathrm{MC}})\mathbf{p}^\top\mathbf{z}^* = (1-\varepsilon_{\mathrm{MC}})\mathrm{OPT}$. ◀

A.3 Proof of Lemma 3.6

Proof. Consider the algorithm **prune** in Figure 9. As we pick a valid amount $r \leq x_S$ to move from x_S to $\check{x}_{\check{S}}$ at each step, $\check{\mathbf{x}}$ must be an ℓ-cover (in the extended set system) when **prune** finishes. Observe that if $\sum_{S:e\in S} x_S < 1$ then e will never be removed from any $\check{S}$,

```
prune(x):
  $\breve{\mathbf{x}} \leftarrow \mathbf{0}_{|\breve{\mathcal{F}}|\times 1}$, $\mathbf{z} \leftarrow \mathbf{0}_{n\times 1}$   ▷ Maintain the pruned solution and its coverage amount
  for each $S \in \mathcal{F}$ do
      $\breve{S} \leftarrow S$
      while $x_S > 0$ do
          $r \leftarrow \min(x_S, \min_{e\in\breve{S}}(1 - z_e))$   ▷ Weight to be moved from $x_S$ to $\breve{x}_{\breve{S}}$
          $x_S \leftarrow x_S - r$, $\breve{x}_{\breve{S}} \leftarrow \breve{x}_{\breve{S}} + r$   ▷ Move weight to the pruned solution
          for each $e \in \breve{S}$ do $z_e \leftarrow z_e + r$   ▷ Update coverage accordingly
          $\breve{S} \leftarrow \breve{S} \setminus \{e \in \breve{S} : z_e = 1\}$   ▷ Remove $e$ with $z_e = 1$ from $\breve{S}$
  return z
```

Figure 9 The **prune** subroutine lifts a solution in $\mathcal{F}$ to a solution in $\breve{\mathcal{F}}$ with the same MaxCover-LP objective value and width 1. The subroutine returns $\mathbf{z}$, the amount by which members of $\breve{\mathcal{F}}$ cover each element. The actual pruned solution $\breve{\mathbf{x}}$ may be computed but has no further use in our algorithm and thus not returned.

so z_e is increased by x_S for every S, and thus $z_e = \sum_{S:e\in S} x_S$. Otherwise, the condition $r \leq 1 - z_e$ ensures that z_e stops increasing precisely when it reaches 1. Each S takes up to $n + 1$ rounds in the while loop as one element $e \in S$ is removed at the end of each round. There are at most m sets, so the algorithm must terminate (in polynomial time).

We note that in Section 3.4, we need to adjust **prune** to instead achieves the condition $z_e = \min(\mathbf{A}_e\mathbf{x}, 1)$ where entries of $\mathbf{A}$ are arbitrary non-negative values. We simply make the following modifications: choose $r \leftarrow \min(x_S, \min_{e\in\breve{S}} \frac{1-z_e}{A_{e,S}})$ and update $z_e \leftarrow z_e + r \cdot A_{e,S}$, and the same proof follows. ◀

Remark that to update the weights in the MWU framework, it is sufficient to have the coverage $\sum_{\breve{S}\in\breve{\mathcal{F}}:e\in\breve{S}} \breve{x}_{\breve{S}}$, which are the z_e's returned by **prune**; the actual solution $\breve{\mathbf{x}}$ is not necessary. Observe further that our MWU algorithm can still use $\mathbf{x}$ instead of $\breve{\mathbf{x}}$ as its solution because $\mathbf{x}$ has no worse coverage than $\breve{\mathbf{x}}$ in every iteration, and so does the final, average solution. Lastly, notice that the coverage $\mathbf{z}$ returned by **prune** has the simple formula $z_e = \min(\sum_{S:e\in S} x_S, 1)$. That is, we introduce **prune** to show an existence of $\breve{\mathbf{x}}$, but will never run **prune** in our algorithm.

A.4 Proof of Lemma 3.8

Proof. Consider the $MaxCover - LP(\mathcal{U}, \mathcal{F}, \ell, \mathbf{p})$ with optimal solution $(\mathbf{x}^{\mathrm{OPT}}, \mathbf{z}^{\mathrm{OPT}})$ of value OPT, and let $\mathbf{x}^{\mathrm{SOL}}$ be a $(1 - \varepsilon_{\mathrm{MC}})$-approximate $\Theta(\frac{\gamma^2}{\log n})$-integral ℓ-cover over the sampled elements and $\mathbf{z}^{\mathrm{SOL}}$ be its corresponding coverage vector. Denote the sampled elements with $\mathcal{L} = \{\hat{e}_1, \cdots, \hat{e}_s\}$. Observe that by defining each X_i as a random variable that takes the value $z^{\mathrm{OPT}}_{\hat{e}_i}$ with probability $p_{\hat{e}_i}$ and 0 otherwise, the expected value of $\mathsf{X} = \sum_{i=1}^{s} \mathsf{X}_i$ is

$$\mathbf{E}[\mathsf{X}] = \sum_{i=1}^{s} \mathbf{E}[\mathsf{X}_i] = s \sum_{e\in\mathcal{U}} p_e \cdot z_e^{\mathrm{OPT}} = s \cdot \mathcal{C}(\mathbf{x}^{\mathrm{OPT}}) = s \cdot \mathrm{OPT}.$$

Let $\tau = s(1 - \gamma)\mathrm{OPT}$. Since $\mathsf{X}_i \in [0, 1]$, by applying Chernoff bound on X, we obtain

$$\begin{aligned}\mathbf{Pr}\left[\mathcal{C}_{\mathcal{L}}(\mathbf{x}^{\mathrm{OPT}}) \leq \tau\right] &= \mathbf{Pr}[\mathsf{X} \leq (1-\gamma)\mathbf{E}[\mathsf{X}]] \\ &\leq e^{-\frac{\gamma^2 \mathbf{E}[\mathsf{X}]}{3}} \leq e^{-\frac{\Omega(\ell \log(mn) \log n/\gamma^2)}{3}} = (mn)^{-\Omega(\ell \log n/\gamma^2)}.\end{aligned}$$

Therefore, since $\mathbf{x}^{\text{SOL}}$ is a $(1-\varepsilon_{\text{MC}})$-approximate solution of $MaxCover - LP(\mathcal{L}, \mathcal{F}, \ell, \mathbf{p})$, with probability $1-(mn)^{-\Omega(\ell \log n/\gamma^2)}$, we have $\mathcal{C}_{\mathcal{L}}(\mathbf{x}^{\text{SOL}}) \geq (1-\varepsilon_{\text{MC}})\tau$.

Next, by a similar approach, we show that for any fractional solution $\mathbf{x}$, if $\mathcal{C}_{\mathcal{L}}(\mathbf{x}) \geq \mathcal{C}_{\mathcal{L}}(\mathbf{x}^{\text{OPT}})$, then with probability $1-(mn)^{-\Omega(\ell \log n/\gamma^2)}$, $\mathcal{C}(\mathbf{x}) \geq \left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}})\text{OPT}$. Consider a fractional ℓ-cover $(\mathbf{x}, \mathbf{z})$ whose coverage is less than $\left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}})\text{OPT}$. Let Y_i denote a random variable that takes value $z_{\hat{e}_i}$ with probability $p_{\hat{e}_i}$, and define $\mathsf{Y} = \sum_{i=1}^{s} \mathsf{Y}_i$. Then, $\mathbf{E}[\mathsf{Y}_i] = \mathcal{C}(\mathbf{x}) < \left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}})\text{OPT}$. For ease of analysis, let each $\bar{\mathsf{Y}}_i \in [0,1]$ be an auxiliary random variable that stochastically dominates Y_i with expectation $\mathbf{E}[\bar{\mathsf{Y}}_i] = \left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}})\text{OPT}$, and $\bar{\mathsf{Y}} = \sum_{i=1}^{s} \bar{\mathsf{Y}}_i$ which stochastically dominates $\bar{\mathsf{Y}}$ with expectation $\mathbf{E}[\bar{\mathsf{Y}}] = s \cdot \left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}})\text{OPT} = \frac{(1-\varepsilon_{\text{MC}})\tau}{1+\gamma}$. We then have

$$\begin{aligned}\mathbf{Pr}[\mathcal{C}_{\mathcal{L}}(\mathbf{x}) > (1-\varepsilon_{\text{MC}})\tau] &= \mathbf{Pr}[\mathsf{Y} > (1-\varepsilon_{\text{MC}})\tau] = \mathbf{Pr}\big[\mathsf{Y} > (1+\gamma)\mathbf{E}[\bar{\mathsf{Y}}]\big] \\ &\leq \mathbf{Pr}\big[\bar{\mathsf{Y}} > (1+\gamma)\mathbf{E}[\bar{\mathsf{Y}}]\big] \leq e^{-\frac{\gamma^2 \mathbf{E}[\bar{\mathsf{Y}}]}{3}} \leq (mn)^{-\Omega(\ell \log n/\gamma^2)},\end{aligned}$$

using the fact that $\left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}}) = \Theta(1)$ for our interested range of parameters. Thus,

$$\mathbf{Pr}\left[\mathcal{C}(\mathbf{x}) \leq \left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}})\text{OPT and } \mathcal{C}_{\mathcal{L}}(\mathbf{x}) > (1-\varepsilon_{\text{MC}})\tau\right] \leq (mn)^{-\Omega(\ell \log n/\gamma^2)}.$$

In other words, except with probability $(mn)^{-\Omega(\ell \log n/\gamma^2)}$, a chosen solution $\mathbf{x}$ that offers at least as good empirical coverage over $\mathcal{L}$ as $\mathbf{x}^{\text{OPT}}$ (namely $\mathbf{x}^{\text{SOL}}$) does have actual coverage of at least $\left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}})\text{OPT}$.

Since the total number of $\Theta(\frac{\gamma^2}{\log n})$-integral ℓ-covers is $O(m^{\ell \log n/\gamma^2})$ (Observation 3.7), applying union bound, with probability at least $1 - O(m^{\ell \log n/\gamma^2}) \cdot (mn)^{-\Omega(\ell \log n/\gamma^2)} = 1 - \frac{1}{\text{poly}(mn)}$, a $(1-\varepsilon_{\text{MC}})$-approximate $\Theta(\frac{\gamma^2}{\log n})$-integral solution of $\mathsf{Max}\,k\text{-}\mathsf{Cover}(\mathcal{L}, \mathcal{F}, \ell, \mathbf{p})$ has weighted coverage of at least $\left(\frac{1-\gamma}{1+\gamma}\right)(1-\varepsilon_{\text{MC}})\text{OPT} > (1-2\gamma)(1-\varepsilon_{\text{MC}})\text{OPT}$ over $\mathcal{U}$. ◀

A.5 Proof of Theorem 3.9

Proof. The algorithm clearly requires $\Theta(T)$ passes to simulate the MWU algorithm. The required amount of memory, besides $\widetilde{O}(n)$ for counting elements, is dominated by the projected set system. In each pass over the stream, we sample $\Theta(\ell \log mn \log n/\varepsilon^4)$ elements, and since they are rarely occurring, each is contained in at most $\Theta(\frac{m}{\varepsilon\ell})$ sets. Finally, we run $\log_{1+\Theta(\varepsilon)} n = O(\log n/\varepsilon)$ instances of the MWU algorithm in parallel to compute a $(1+\varepsilon)$-approximate solution. In total, our space complexity is $\Theta(\ell \log mn \log n/\varepsilon^4) \cdot \Theta(\frac{m}{\varepsilon\ell}) \cdot O(\log n/\varepsilon) = \widetilde{O}(m/\varepsilon^6)$. ◀

A.6 Proof of Lemma 3.10

Proof. Recall the weight update formula $w_e^{t+1} = w_e^t(1 - \frac{\beta(\breve{\mathbf{A}}_e \breve{\mathbf{x}} - b_e)}{6\phi})$ for the MWU framework, where $\breve{\mathbf{A}}_{n \times |\breve{\mathcal{F}}|}$ represents the membership matrix corresponding to the extended set system $(\mathcal{U}, \breve{\mathcal{F}})$. In our case, the desired coverage amount is $b_e = 1$. By construction, we have $\breve{\mathbf{A}}_e \breve{\mathbf{x}} = z_e \leq 1$; therefore, our width is $\phi = 1$, and $-1 \leq \breve{\mathbf{A}}_e \breve{\mathbf{x}} - b_e \leq 0$. That is, the weight of each element cannot decrease, but may increase by at most a multiplicative factor of $1 + \beta/6$, before normalization. Thus even after normalization no weight may increase by more than a factor of $1 + \beta/6 = 1 + O(\beta)$. ◀

Online Strip Packing with Polynomial Migration*†

Klaus Jansen[1], Kim-Manuel Klein[2], Maria Kosche[3], and Leon Ladewig[4]

1 **Department of Computer Science, Kiel University, Kiel, Germany**
kj@informatik.uni-kiel.de

2 **Department of Computer Science, Kiel University, Kiel, Germany**
kmk@informatik.uni-kiel.de

3 **Department of Computer Science, Kiel University, Kiel, Germany**
mkos@informatik.uni-kiel.de

4 **Department of Computer Science, Technical University of Munich, Munich, Germany**
ladewig@in.tum.de

Abstract

We consider the relaxed online strip packing problem, where rectangular items arrive online and have to be packed into a strip of fixed width such that the packing height is minimized. Thereby, repacking of previously packed items is allowed. The amount of repacking is measured by the migration factor, defined as the total size of repacked items divided by the size of the arriving item. First, we show that no algorithm with constant migration factor can produce solutions with asymptotic ratio better than 4/3. Against this background, we allow amortized migration, i. e. to save migration for a later time step. As a main result, we present an AFPTAS with asymptotic ratio $1 + \mathcal{O}(\epsilon)$ for any $\epsilon > 0$ and amortized migration factor polynomial in $1/\epsilon$. To our best knowledge, this is the first algorithm for online strip packing considered in a repacking model.

1998 ACM Subject Classification G.2.1 Combinatorial algorithms

Keywords and phrases strip packing, bin packing, online algorithms, migration factor

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.13

1 Introduction

In the classical *strip packing* problem we are given a set of two-dimensional items with heights and widths bounded by 1 and a strip of infinite height and width 1. The goal is to find a packing of all items into the strip without rotations such that no items overlap and the height of the packing is minimal. In many practical scenarios, the entire input is not known in advance. Therefore, an interesting field of study is the *online* variant of the problem. Here, items arrive over time and have to be packed immediately without knowing future items. Following the terminology of [11] for the online bin packing problem, in the *relaxed online strip packing* problem previous items may be repacked when a new item arrives.

There are different ways to measure the amount of repacking in a relaxed online setting. We follow the *migration model* introduced by Sanders, Sivadasan, and Skutella in [24]. For online job scheduling on identical parallel machines they defined the *migration factor* μ as

[*] A full version of the paper is available at http://arxiv.org/abs/1706.04939.

[†] This work was partially supported by DFG Project, "Robuste Online-Algorithmen für Scheduling- und Packungsprobleme", JA 612 /19-1, and by GIF-Project "Polynomial Migration for Online Scheduling".

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 13; pp. 13:1–13:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

follows: When a new job of size p_j arrives, jobs of total size μp_j can be reassigned. In the context of online strip packing the migration factor μ ensures that the total area of repacked items is at most μ times the area of the arrived item.

By a well known relation between strip packing and parallel job scheduling [14], any (online) strip packing algorithm applies to (online) scheduling of parallel jobs. The latter problem is highly relevant e. g. in computer systems [14, 27, 23].

Preliminaries. Since strip packing is NP-hard [1], research focuses on efficient approximation algorithms. Let $A(I)$ denote the packing height of algorithm A on input I and $\mathrm{OPT}(I)$ the minimum packing height. The *absolute (approximation) ratio* is defined as $\sup_I A(I)/\mathrm{OPT}(I)$ while the *asymptotic (approximation) ratio* as $\limsup_{\mathrm{OPT}(I)\to\infty} A(I)/\mathrm{OPT}(I)$. Typically, the performance of online algorithms is measured by *competitive analysis*, where an online algorithm is compared with an optimal offline algorithm. In the following, all ratios mentioned in the context of online algorithms are competitive.

1.1 Related Work

Offline. Strip packing is one of the classical packing problems and receives ongoing research interest in the field of combinatorial optimization. Since Baker, Coffman and Rivest [1] gave the first algorithm with asymptotic ratio 3, strip packing was investigated in many studies, considering both asymptotic and absolute approximation ratios. We refer the reader to [6] for a survey. For the asymptotic ratio, a well-known result is the AFPTAS by Kenyon and Rémila [21]. Concerning the absolute ratio, currently the best known algorithm of ratio $5/3+\epsilon$ is by Harren et al. [13].

An interesting result was given by Han et al. in 2007. In [12], they studied the relation between bin packing and strip packing and developed a framework between both problems. For the offline case, it is shown that any bin packing algorithm can be applied to strip packing while maintaining the same asymptotic ratio.

Online. The first algorithm for online strip packing was given by Baker and Schwarz [2] in 1983. Using the concept of shelf algorithms [1], they derived the algorithm *First-Fit-Shelf* with asymptotic ratio arbitrary close to 1.7 and absolute ratio 6.99 (where all rectangles have height at most $h_{\max} = 1$). Later, Csirik and Woeginger [8] showed a lower bound of $h_\infty \approx 1.69$ on the asymptotic ratio for the concept of shelf algorithms and gave an improved shelf algorithm with asymptotic ratio $h_\infty + \epsilon$ for any $\epsilon > 0$. The framework of Han et al. [12] is applicable in the online setting if the online bin packing algorithm belongs to the class *Super Harmonic*. Using Seiden's online bin packing algorithm *Harmonic++* [25], there exists an algorithm for online strip packing with an asymptotic ratio of 1.58889. In 2007 and 2009, the concept of *First-Fit-Shelf* by Baker and Schwarz was improved independently by two research groups, Hurink and Paulus [14] and Ye, Han, and Zhang [28]. Both improve the absolute competitive ratio of from 6.99 to 6.6623 without a restriction on $h_{\max}$. Further results on special variants of online strip packing were given by Imeh [16] and Ye, Han, and Zhang [29].

On the negative side, there is no algorithm for online strip packing (without repacking) with an asymptotic ratio better then 1.5404 since the lower bound in [3] for online bin packing is also valid for online strip packing. Regarding the absolute ratio, the first lower bound of 2 from [5] was improved in several studies [19, 15, 22]. Currently, the best known lower bound by Yu, Mao, and Xiao [30] is $(3+\sqrt{2})/2 \approx 2.618$.

Related results on the migration model. Since its introduction by Sanders, Sivadasan, and Skutella [24], the migration model became increasingly popular. In the context of online scheduling on identical machines, Sanders, Sivadasan, and Skutella [24] gave a PTAS with migration factor $2^{\mathcal{O}\left((1/\epsilon)\log^2(1/\epsilon)\right)}$ for the objective of minimizing the makespan. Thereby, the migration factor in [24] depends only on the approximation ratio ϵ and not on the input size. Such algorithms are called *robust*.

Skutella and Verschae [26] studied scheduling on identical machines while maximizing the minimum machine load, called *machine covering*. They considered the *fully dynamic* setting in which jobs are also allowed to depart. Due to the presence of very small jobs, Skutella and Verschae showed that there is no PTAS for this problem in the migration model. Instead, they introduced the *reassignment cost model*, in which an amortized analysis of the migration factor is allowed. Using the reassignment cost model, they gave a robust PTAS for the problem with amortized migration factor $2^{\mathcal{O}\left((1/\epsilon)\log^2(1/\epsilon)\right)}$.

Also online bin packing has been investigated in the migration model in a sequence of papers, inspired by the work of Sanders, Sivadasan, and Skutella [24]: The first robust APTAS for relaxed online bin packing was given in 2009 by Epstein and Levin [10]. They obtained an exponential migration factor $2^{\mathcal{O}\left((1/\epsilon^2)\log 1/\epsilon\right)}$. In 2013, Jansen and Klein [17] improved this result and gave an AFPTAS with polynomial migration factor $\mathcal{O}\left(\frac{1}{\epsilon^3}\log\frac{1}{\epsilon^4}\right)$. The development of advanced LP/ILP-techniques made this notable improvement possible. Furthermore, in [4] Berndt, Jansen, and Klein used the techniques developed in [17] to give an AFPTAS for fully dynamic bin packing with a similar migration factor.

Our contribution

To the authors knowledge, there exists currently no algorithm for online strip packing in the migration or any other repacking model. Therefore, we present novel ideas to obtain the following results: First, a relatively simple argument shows that in the (strict) migration model it is not possible to maintain solutions that are close to optimal. We prove the following theorem in Section 1.3:

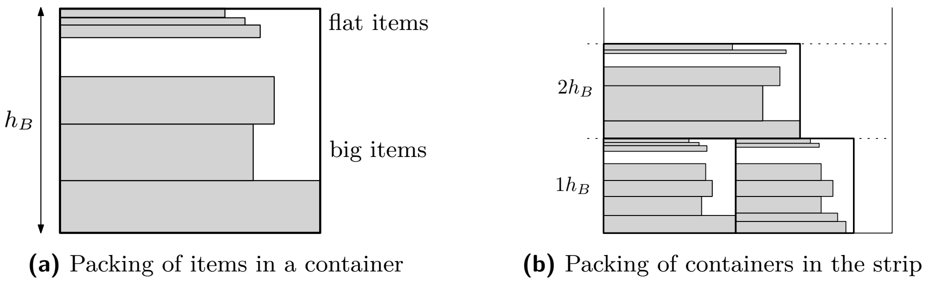
▶ **Theorem 1.1.** *In the (strict) migration model, there is no approximation algorithm for relaxed online strip packing with asymptotic competitive ratio better than* $4/3$*.*

For this reason, it is natural to extend the migration model such that amortization is allowed. We say that an algorithm has an *amortized* migration factor of μ if for every time step t, the total migration (i. e. the total area of repacked items) up to time t is bounded by $\mu\sum_{i=1}^{t}\mathrm{SIZE}(i_t)$, where $\mathrm{SIZE}(i_t)$ is the area of the item arrived at time t. Adapted to scheduling problems, this corresponds with the reassignment cost model introduced by Skutella and Verschae in [26]. Consequently, we focus on an approach that makes use of amortization and therefore admits an asymptotic approximation scheme. We adapt several offline and online techniques and combine them with our novel approaches to obtain the following main result:

▶ **Theorem 1.2.** *There is a robust AFPTAS for relaxed online strip packing with an amortized migration factor polynomial in* $1/\epsilon$*.*

1.2 Technical Contribution

A general approach in the design of robust online algorithms is to rethink existing algorithmic strategies that work for the corresponding offline problem in a way that the algorithm can



(a) Packing of items in a container **(b)** Packing of containers in the strip

Figure 1 Packing structure of our approach. Items are packed into containers of fixed height h_B, thus the packing of containers results in a bin packing problem.

adapt to a changing problem instance. The experiences that were made so far in the design of robust algorithms (see [17, 4, 26]) are to design the algorithm in a way such that the generated solutions fulfill very tight structural properties. Such solutions can then be adapted more easily as new items arrive.

A first approach would certainly be do adapt the well known algorithm for (offline) strip packing by Kenyon and Rémila [21] to the online setting. However, we can argue that the solutions generated by this algorithm do not fulfill sufficient structural properties. In the algorithm by Kenyon and Rémila, the strip is divided vertically into segments, where each segment is configured with a set of items. Thereby, each segment can have a different height. Now consider the online setting, where we are asked for a packing for the enhanced instance that maintains most parts of the existing packing. Obviously, it is not enough to place new items on top of the packing as this would exceed the approximation guarantee. To guarantee a good competitive ratio, existing configurations of the segments need to be changed. However, this seems to be hard to do as the height of a configuration can change. Gaps can occur in the packing as a segment might decrease in height or vice versa a segment might increase in height and therefore does not fit anymore in its current position. Over time this can lead to a very fragmented packing. On the other hand, closing gaps in a fragmented packing can cause a huge amount of repacking.

Containers. Therefore, we follow a different approach to develop an algorithm that guarantees solutions with a more modular structure. A central idea is to batch items to larger rectangles of fixed height, called *containers* (see Figure 1a). As each container has the same height h_B, it is natural to divide the strip into *levels* of equal height h_B (see Figure 1b) and fill each level with containers best possible. Thus, finding a container packing is in fact a bin packing problem, where levels correspond with bins and the sizes of the bin packing items are given by the container widths. This approach was studied in the offline setting by Han et al. in [12], while an analysis in the online setting is more sophisticated.

Thus, the packing of items into the strip is given by two assignments: By the *container assignment* each item is assigned a container where its is placed. Moreover, the *level assignment* describes which container is placed in which level (corresponds with the bin packing solution). To guarantee solutions with good approximation ratio, both functions have to satisfy certain properties.

Dynamic rounding / Invariant properties. For the container assignment, a natural choice would certainly be to assign the widest items to the first container, the second widest to the second container, and so on. In [12], Han et al. show that this container assignment is

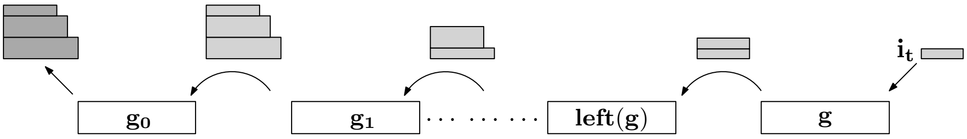


Figure 2 Shift operation moves widest items between groups to insert new item i_t.

somehow optimal. However, in the online setting we can not maintain this strict order while bounding the repacking size. Therefore, we use a relaxed ordering by introducing groups for containers of similar width and requiring the sorting over the groups, rather than over containers. For this purpose, we adapt the dynamic rounding technique developed by Berndt, Jansen, and Klein in [4] and formulate important characteristics as *invariant properties*.

Shift. In order to insert new items, we develop an operation called Shift. The idea is to move items between containers of different groups such that the invariant properties stay fulfilled. When inserting an item i_t via Shift into group[1] g, items are moved from g to the group *left*(g), where again items are shifted to the next group, and so on (see Figure 2). Thereby, the total height of the shifted items can increase in each step. However, it is limited such that items that can not be shifted further (at group g_0 in Figure 2) can be packed into one additional container. This way, we get a new container assignment for the enhanced instance which maintains the approximation guarantee and all desired structural properties.

LP/ILP-techniques. As a consequence of the shift operation, there may be a new container which has to be inserted into the packing. Obviously, placing new containers always into new levels may lead to a level assignment which does not satisfy the approximation guarantee. Therefore, the existing level assignment has to be changed, which causes further repacking. We apply the LP/ILP-techniques developed in [17] to maintain a good level assignment while the amortized migration factor is polynomial in $1/\epsilon$.

Packing of small items. Another challenging part regards the handling of items with small area. Without maintaining an advanced structure, small items can fractionate the packing in a difficult way. Such difficulties also arise in related optimization problems, see e.g. [26, 4]. For the case of flat items (with small height) we overcome these difficulties by the packing structure shown in Figure 1a: Flat items are separated from big items in the containers and are sorted by width such that the least wide item is at the top. Narrow items (small width) can be used to fill gaps in the packing while grouping narrow items of similar height. We sketch some ideas for the packing of small items in the Appendix A-B and refer to the full version [18] for all details.

1.3 Lower Bound

In this section we prove Theorem 1.1. We use an adversary to construct an instance with arbitrary optimal packing height, but $A(I) \geq \frac{4}{3}\,\mathrm{OPT}(I)$ for any such algorithm A.

Proof. Let A be an algorithm for relaxed online strip packing with migration factor μ. We show that for any height h there is an instance I with $\mathrm{OPT}(I) \geq h$ and $A(I) \geq \frac{4}{3}\,\mathrm{OPT}(I)$. The instance consists of two item types: A *big* item has width $\frac{1}{2} - \epsilon$ and height 1, while a *flat*

[1] In the following, by 'group of an item' we mean the group of the container in which the item is placed.

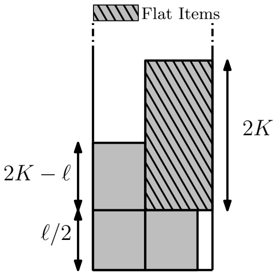


Figure 3 Optimal online packing.

item has width $\frac{1}{2} + \epsilon$ and height $\frac{1}{2\lceil\mu\rceil}$. For an item i let SIZE(i) denote its area. Note that A can not repack a big item b when a flat item f arrives, as SIZE$(b) > \mu\,$SIZE(f) for $\epsilon < 1/6$.

First, the adversary sends $2K$ big items, where $K = 3\lceil h\rceil$. Let ℓ be the number of big items that are packed by A next to another big item. The packing has a height of at least $\frac{\ell}{2} + 2K - \ell = 2K - \frac{\ell}{2}$ (see Figure 3). Since the optimum packing height for $2K$ big items is K (always two items in one level), A has an absolute ratio of at least $2 - \frac{\ell}{2K}$. If $\ell \leq \frac{4K}{3}$, the absolute ratio is at least $\frac{4}{3}$ and nothing else is to show.

Now assume $\ell > \frac{4K}{3}$. In this case, the adversary sends $k = 4\lceil\mu\rceil K$ flat items of total height $2K$. In the optimal packing of height $2K$ big items and flat items form separate stacks that are placed next to each other. Note that no two flat items can be packed next to each other. Since A can not repack any big item when a flat item arrives, in the best possible packing achievable by A flat items of total height $2K - \ell$ are packed next to big items (see Figure 3, flat items are packed in the dashed area). Therefore, the packing height is at least $2K + \frac{\ell}{2}$ and hence the absolute ratio is at least $1 + \frac{\ell}{4K} \geq \frac{4}{3}$. In either case, it follows that the asymptotic ratio is at least $4/3$ by considering $K \to \infty$. ◀

1.4 Remainder of the Paper

In the remainder of this paper we give a high-level description of the proof of Theorem 1.2. Thereby, we focus on *big items* having minimum area ϵ^2 (see below). For most of the technical details as well as the handling of small items we refer to the full version [18].

Throughout the following sections, let $\epsilon \in (0, 1/4]$ be a constant such that $1/\epsilon$ is integer. We denote the height and width of an item i by $h(i)$ and $w(i)$ (both at most 1) and define SIZE$(i) = w(i)h(i)$. An item i is called *big* if $h(i) \geq \epsilon$ and $w(i) \geq \epsilon$. Let I_L be the set of big items. If R is a set of items, let SIZE$(R) = \sum_{i\in R}$ SIZE(i) and $h(R) = \sum_{i\in R} h(i)$.

2 Container Packing

Recall that we follow a two-level-approach to obtain the actual packing: Items are packed into containers of equal height h_B, whereby the widest item inside a container defines its width (see Figure 1a). The strip is divided into levels of height h_B, where the containers are packed (see Figure 1b). In this section we state important *invariant properties* concerning the relation between items and containers. Further, we show that if these invariant properties hold, the container packing yields a good approximation to the strip packing problem.

In order to find a container packing, we use a common LP formulation by Eisemann [9] (see also [4, 17]). However, the number of occurring container widths has to be bounded to solve the LP efficiently. Therefore, we introduce groups for the containers and round each container width to the largest width in its group, similar to rounding techniques in bin packing [20]. Nevertheless, the rounding has to be flexible enough for the online setting. We

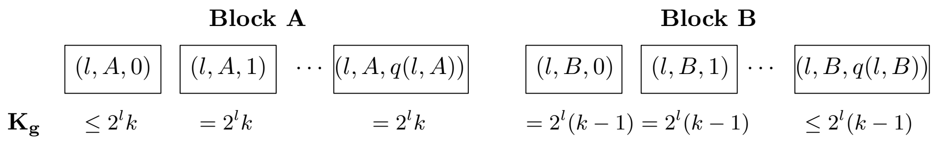


Figure 4 Groups of one category $l \in W$ and number of containers K_g per group.

adapt the dynamic rounding technique developed in [4], which is described in the following section.

2.1 Dynamic Rounding

Let $\mathcal{C}$ be the set of containers. Each container is assigned to a *(width) category* $l \in \mathbb{N}$, where container c has width category l if $w(c) \in (2^{-(l+1)}, 2^{-l}]$. Let W denote the set of all non-empty categories and define $\omega = |W|$ in the following. It follows immediately that $\omega = \mathcal{O}(\log 1/\epsilon)$. Furthermore, we build groups within the categories: A group $g \in G$ is a triple (l, X, r), where $l \in W$ is the category, $X \in \{A, B\}$ is the *block*, and $r \in \mathbb{N}$ is the *position* in the block. The maximum position of category l at block X that is non-empty is denoted by $q(l, X)$. Figure 4 outlines the group structure of one category $l \in W$ (the values for K_g will become clear in Section 2.2). For a group $g = (l, X, r)$ the groups *left*(g) and *right*(g) are defined as the respective neighboring groups[2] in the order shown in Figure 4.

By the notion of blocks, groups of one category can be partitioned into two types. This becomes helpful to maintain the invariant properties with respect to the growing set of items. More details on that are given in the later Sections 2.2 and 3.2.

The assignment from containers to groups is given by a *rounding function* $R \colon \mathcal{C} \to G$. Let $K_g = |\{c \in \mathcal{C} \mid R(c) = g\}|$ be the number of containers of group g. Let ${I_L}^g$ be the set of items in (containers of) group g.

2.2 Invariant Properties

In Section 1.2 we argued that only solutions with strong structural properties can be adapted appropriately in the online setting while maintaining a good competitive ratio. Definition 2.1 formalizes this central properties.

▶ **Definition 2.1** (Invariant properties). Let $k \in \mathbb{N}$ be a parameter and $h(g) = \sum_{i \in {I_L}^g} h(i)$ be the total height of items in group g.

(a) Items correspond to categories
$2^{-(l+1)} < w(i) \leq 2^{-l} \qquad \forall i \in {I_L}^g \text{ s.t. } g = (l, \cdot, \cdot)$

(b) Sorting of items over groups
$w(i) \geq w(i') \qquad \forall i \in {I_L}^g, i' \in {I_L}^{g'} \text{ s.t. } g = \mathit{left}(g')$

(c) Number of containers in block A
$K_{(l,A,0)} \leq 2^l k,$
$K_{(l,A,r)} = 2^l k \qquad \forall l \in W \text{ and } \forall 1 \leq r \leq q(l, A)$

(d) Number of containers in block B
$K_{(l,B,q(l,B))} \leq 2^l (k-1),$
$K_{(l,B,r)} = 2^l (k-1) \qquad \forall l \in W \text{ and } \forall 0 \leq r < q(l, B)$

(e) Total height of items per group
$(h_B - 1)(K_g - 1) \leq h(g) \leq (h_B - 1) K_g \qquad \forall g \in G$

[2] Set *left*$((l, A, 0)) = (l, A, -1)$ and *right*$((l, B, q(l, B))) = (l, B, q(l, B) + 1)$ as temporary groups.

Property (a) ensures that each item is inserted into the right category. Note that as a consequence, each container of a group $(l,\cdot,\cdot)$ has a width in $(2^{-(l+1)}, 2^{-l}]$. By property (b), all items in a group g have a width greater or equal than items in the group *right*(g). That is, instead of a strict order over all containers, (b) ensures an order over groups of containers. The properties (c) and (d) set the number of containers to a fixed value, except for special cases (see Figure 4): Groups in block A have more containers than groups in block B. Moreover, there are two *flexible groups* (namely $(l, A, 0)$ and $(l, B, q(l, B))$) whose number of containers is only upper bounded. Finally, property (e) ensures an important relation between items and containers of one group g: Since $h(g) \leq K_g(h_B - 1)$, at least one of the K_g containers has a filling height of at most $h_B - 1$ and thus can admit a new item. However, the lower bound $(h_B - 1)(K_g - 1) \leq h(g)$ ensures that each container is well filled in an average container assignment.

One of the important consequences of the invariant properties is the fact that the number of non-empty groups $|G|$ can be bounded from above, assuming that the instance is not too small. Therefore, the parameter k has to be set in a particular way:

▶ **Lemma 2.2.** *For* $k = \left\lfloor \frac{\epsilon}{4\omega h_B} \operatorname{SIZE}(I_L) \right\rfloor$ *the number of non-empty groups in* G *is bounded by* $\mathcal{O}\left(\frac{\omega}{\epsilon}\right) = \mathcal{O}\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)$, *assuming that* $\operatorname{SIZE}(I_L) \geq \frac{24\omega h_B(h_B-1)}{\epsilon h_B - 2\epsilon}$.

Proof. Let $G_1 = G \setminus \left(\bigcup_{l\in W}(l, A, 0) \cup \bigcup_{l\in W}(l, B, q(l, B))\right)$ and let $g \in G_1$. Since by invariant (a) every container of group g has width greater than $2^{-(l+1)}$, it follows together with the further invariant properties

$$\begin{aligned}
\operatorname{SIZE}(I_L{}^g) &> 2^{-(l+1)}(h_B - 1)(K_g - 1) && \text{(a), (e)}\\
&\geq 2^{-(l+1)}(h_B - 1)(2^l(k-1) - 1) && \text{(c), (d)}\\
&= \frac{1}{2}(h_B - 1)(k-1) - 2^{-(l+1)}(h_B - 1)\\
&\geq \frac{1}{2}(h_B - 1)(k-1) - \frac{h_B - 1}{2}\\
&= \frac{1}{2}(h_B - 1)(k-2)\,.
\end{aligned}$$

Now, let $I_L^{(l)}$ be the set of items in I_L which belong to containers of category l. It holds that $\operatorname{SIZE}(I_L^{(l)}) \geq \sum_{g=(l,\cdot,\cdot)\in G_1} \operatorname{SIZE}(I_L{}^g) \geq (q(l, A) + q(l, B))\left(\frac{1}{2}(h_B - 1)(k-2)\right)$ and resolving leads to

$$q(l, A) + q(l, B) \leq \frac{2\operatorname{SIZE}(I_L^{(l)})}{(h_B - 1)(k-2)}\,. \tag{1}$$

We now show $(h_B - 1)(k-2) \geq \frac{\epsilon}{8\omega h_B} \operatorname{SIZE}(I_L)$. The assumption on $\operatorname{SIZE}(I_L)$ is equivalent to $\frac{\epsilon}{4\omega h_B}\operatorname{SIZE}(I_L) - 3 \geq \frac{\epsilon}{8\omega(h_B-1)}\operatorname{SIZE}(I_L)$. Therefore,

$$k - 2 = \left\lfloor \frac{\epsilon}{4\omega h_B}\operatorname{SIZE}(I_L)\right\rfloor - 2 \geq \frac{\epsilon}{4\omega h_B}\operatorname{SIZE}(I_L) - 3 \geq \frac{\epsilon}{8\omega(h_B - 1)}\operatorname{SIZE}(I_L)$$

and thus

$$(h_B - 1)(k-2) \geq \frac{(h_B - 1)\epsilon}{8\omega(h_B - 1)}\operatorname{SIZE}(I_L) = \frac{\epsilon}{8\omega}\operatorname{SIZE}(I_L)\,.$$

Further, we get

$$\frac{2\operatorname{SIZE}(I_L)}{(h_B - 1)(k-2)} \leq \frac{2\operatorname{SIZE}(I_L)}{\frac{\epsilon}{8\omega}\operatorname{SIZE}(I_L)} = \frac{16\omega}{\epsilon}\,. \tag{2}$$

As shown in Figure 4, for each category l there are $q(l,A)+q(l,B)+2$ groups. Now, summing over all categories $l \in W$ concludes the proof:

$$\begin{aligned}
&\sum_{l\in W} q(l,A)+q(l,B)+2 \\
&\leq \sum_{l\in W}\left(\frac{2\,\mathrm{SIZE}(I_L^{(l)})}{(h_B-1)(k-2)}+2\right) && \text{eq. (1)}\\
&= 2\,|W| + \frac{2}{(h_B-1)(k-2)}\sum_{l\in W}\mathrm{SIZE}(I_L^{(l)})\\
&= 2\,|W| + \frac{2\,\mathrm{SIZE}(I_L)}{(h_B-1)(k-2)}\\
&\leq 2\omega + \frac{16\omega}{\epsilon} && \text{eq. (2)}
\end{aligned}$$

◀

2.3 Approximation Guarantee

Furthermore, we can argue that if the invariant properties of Definition 2.1 are fulfilled, the rounded container packing yields a good approximation to a packing of the instance I_L.

Let $con\colon I_L \to \mathcal{C}$ be a container assignment and $R\colon \mathcal{C} \to G$ be a rounding function fulfilling the invariant properties (a-e). Formally, we define the rounded container instance C^R_{con} as follows: For each container $c \in \mathcal{C}$ such that there exists an item i with $con(i) = c$, define a rectangle of height $h(c) = h_B$ and width $w(c) = \max\{w(i) \mid i \in I_L, con(i) = c\}$. Then, round each container width to the largest width in its group defined by R.

By choosing $h_B = 13/\epsilon^2$ and $k = \left\lfloor \frac{\epsilon}{4\omega h_B}\,\mathrm{SIZE}(I_L)\right\rfloor$ as parameters of the invariant, we get the following result:

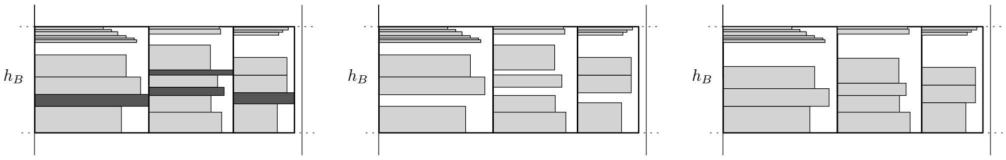
▶ **Lemma 2.3.** *Let C^R_{con} be the strip packing instance of rounded containers fulfilling all invariant properties from Definition 2.1. Assuming* $\mathrm{SIZE}(I_L) \geq \frac{4\omega h_B}{\epsilon}(h_B+1)$*, it holds that* $\mathrm{OPT}(C^R_{con}) \leq (1+4\epsilon)\,\mathrm{OPT}(I_L) + \mathcal{O}\left(1/\epsilon^4\right)$.

Proof (Sketch). In [18] we give a detailed proof using a proof technique from [12]. For the sake of intuition, in this paper we only sketch the main arguments necessary to proof Lemma 2.3. The proof uses a nice combination of all invariant properties from Definition 2.1.

Intuitively, the goal is to show that by packing the containers C^R_{con} instead of the items I_L, we do not loose too much area in the packing. This can be shown formally by defining two sets of rectangles: Let $\hat{I}_L$ be the set of items in I_L where the width of each item from group g is set to the widest item in the group $right(g)$. Note that by invariant (b), the widths of items from $\hat{I}_L$ get rounded down. As the heights remain unchanged, it holds that $\mathrm{OPT}(\hat{I}_L) \leq \mathrm{OPT}(I_L)$. Furthermore, let C_1 be the set of all container rectangles from C^R_{con}, except from the left- and right-most groups of each category l.

For the moment, assume that each container is filled up to the maximum filling height h_B. Therefore, we have a relation between $\hat{I}_L$ and C_1: Each stack of rounded-down rectangles from $\hat{I}_L$ corresponds with a container rectangle from C_1, namely with one of the group to the right. Therefore, packing C_1 instead of $\hat{I}_L$ is basically the same. By invariant (e), the total height of items in each group is bounded from below. Thus we can think of an average container assignment, in which each container is well-filled also in height. Therefore, the packing capacity of each container is used efficiently in height and width.

Finally, we have to argue that the containers dropped from C^R_{con} to obtain C_1 can be packed such that the total packing height increases only by a small term. By invariant (c–d),



(a) Set S_{out} (dark items) **(b)** Gaps after removal of S_{out} **(c)** After SINK

Figure 5 Operation SINK closes gaps during a SHIFT operation.

the left- and right-most groups of a category l have each at most $2^l k$ containers, all of width at most 2^{-l} by invariant (a). That is, $2k$ levels of height h_B are enough to place all residual containers in C^R_{con} not contained in C_1. By definition of k and ω, it follows that the additional packing height for the missing containers in C_1 is not more than $\epsilon\,\mathrm{SIZE}(I_L) \leq \epsilon\,\mathrm{OPT}(I_L)$. ◀

3 Shift Operation

So far, we introduced the packing structure and showed important characteristics of it. In this section we consider the online setting, where new items arrive and have to be integrated into the structure such that invariant properties (a-e) are maintained. In order to maintain (a-b) when inserting a new item i, a suitable group has to be found, defined as follows:

▶ **Definition 3.1** (Suitable group). For a group g, let $w_{min}(g)$ resp. $w_{max}(g)$ denote the width of an item with minimal resp. maximal width in ${I_L}^g$. Set $w_{min}(\mathit{left}((l, A, 0))) = \infty$ and $w_{max}(\mathit{right}((l, B, q(l, B)))) = 0$. Group $g = (l, X, r)$ is *suitable* for a new item i if $w(i) \in (2^{-(l+1)}, 2^{-l}]$, $w_{min}(\mathit{left}(g)) \geq w(i)$, and $w_{max}(\mathit{right}(g)) < w(i)$.

Basically, new items can be integrated into the container structure in two ways: They can be placed into new containers, or they can be placed into existing containers, where already packed items have to be removed possibly.

Since the first option occurs rather in special cases, in Section 3.1 we describe a simplified version of the SHIFT operation which inserts items via the second way. Note that in this case the number of containers remains unchanged and thus (c) and (d) are maintained anyway. Afterwards, we briefly describe the issue of new containers in the packing.

3.1 Shift Algorithm (simplified)

Algorithm 1 shows the (simplified) SHIFT operation. Suppose that S is a set of items to be inserted into the suitable group g. The easy case is when $h(g) + h(S)$ does not exceed the upper bound $(h_B - 1)K_g$ from invariant (e): Then, PLACE(g, S) in Line 3 packs each item in S into any container with sufficient small packing height[3] of this group. It can be easily seen that there must be such a container: Assume that item $i \in S$ can not be placed. Then, each of the K_g containers is filled with items of total height greater than $h_B - 1$. Thus, $h(g) + h(S) > K_g(h_B - 1)$, which contradicts (e).

However, the crucial point is that due to the insertion of S, the total height of items in g could exceed the upper bound from (e). In order to fulfill (e), items from g are removed. For this purpose, we choose the widest items from g, as they can be inserted into the group $\mathit{left}(g)$ while maintaining the sorting property (b). The function WIDESTITEMS$({I_L}^g \cup S, \Delta)$ in Line 5

[3] That is, the total height of items in this container plus the height of the new item does not exceed h_B.

Algorithm 1: Shift

```
  Input : Group $g \in G$, Items $S \subset I_L$, suitable for $g$ according to Definition 3.1
1 $\Delta = h(g) + h(S) - (h_B - 1)K_g$
2 if $\Delta \leq 0$ then                        // No violation of invariant (e)
3     Place($g$,$S$)
4 else
5     $S_{out}$ = WidestItems($I_L{}^g \cup S$, $\Delta$)
6     Remove $S_{out}$ from group
7     Sink($c_j$)              // For all affected containers $c_j$
8     Place($g$,$S$)
9     Shift($left(g), S_{out}$)
```

returns a set of items $S_{out} \subseteq I_L{}^g \cup S$ s.t. $w(i) \geq w(i')$ for each $i \in S_{out}, i' \in (I_L{}^g \cup S) \setminus S_{out}$ and $h(S_{out}) \in [\Delta, \Delta + 1)$. Note that after removing the items S_{out}, gaps may occur in the packing. These have to be closed before new items can be placed, which is done by the operation Sink in Line 7 (see Figures 5a to 5c for an illustration). Now, there is enough room to place the items S in Line 8. The removed items get inserted into *left*(g) via a further Shift operation. If the group *left*(g) does not exist, one has to open a new container for the remaining items.

An important characteristic of Algorithm 1 is that it maintains all invariant properties. In the following we give a proof restricted to property (e), as this is somehow the most fundamental property.

▶ **Lemma 3.2.** *Suppose that invariant property (e) holds. After shifting items S into group g via Algorithm 1, invariant property (e) remains fulfilled.*

Proof. We show that the total height of items after the removal of S_{out} and insertion of S lies in the interval $[(h_B - 1)(K_g - 1), (h_B - 1)K_g]$. Since $h(S_{out}) \geq \Delta$, it holds $h(g) - h(S_{out}) + h(S) \leq h(g) - \Delta + h(S) = h(g) - h(g) - h(S) + (h_B - 1)K_g + h(S) = (h_B - 1)K_g$. On the other side, $h(S_{out}) < \Delta + 1$ and thus $h(g) - h(S_{out}) + h(S) > h(g) - \Delta - 1 + h(S) = h(g) - h(g) - h(S) + (h_B - 1)K_g - 1 + h(S) = (h_B - 1)K_g - 1$. With $h_B \geq 2$ it follows that $(h_B - 1)K_g - 1 \geq (h_B - 1)(K_g - 1)$. Hence, property (e) is fulfilled. ◀

Since the set S_{out} is inserted via another shift operation into the next group, in general the insertion of an item i_t triggers a sequence of shift operations Shift(g^0, S^0), Shift(g^1, S^1), …, Shift(g^d, S^d) with $S^0 = \{i_t\}$. Thereby, the total height of shifted items $h(S_{out})$ grows linearly in the position of the shift sequence, like the following lemma shows.

▶ **Lemma 3.3.** *Consider the above defined shift sequence and let S^j_{out} be the set S_{out} in the call* Shift(g^j, S^j). *For any j with $0 \leq j \leq d$ it holds that $h(S^j_{out}) \leq h(S^0) + j + 1$.*

Proof. Let Δ_j denote the value of Δ in the call Shift(g^j, S^j). First note that by invariant (e) $\Delta_j \leq h(S^j)$ holds for each j . Further, the function WidestItems$(\cdot, \Delta_j)$ returns a set S^j_{out} with $h(S^j_{out}) < \Delta_j + 1$. For $j = 0$ it holds that $h(S^0_{out}) < \Delta_0 + 1 \leq h(S^0) + 1$. Now suppose $h(S^j_{out}) \leq h(S^0) + j + 1$ for some $j \geq 0$. Note that $S^{j+1} = S^j_{out}$, thus for the index $j + 1$ we have $h(S^{j+1}_{out}) < \Delta_{j+1} + 1 \leq h(S^{j+1}) + 1 = h(S^j_{out}) + 1$. By assumption, $h(S^j_{out}) \leq h(S^0) + j + 1$ and thus $h(S^{j+1}_{out}) \leq h(S^0) + (j + 1) + 1$. ◀

Lemma 3.3 is particularly important to bound the amount of items arriving in the leftmost group. By choosing h_B appropriately, the remaining items fit into one additional container.

Algorithm 2: Insertion of a big item

```
Input: Item $i_t \in I_L$
if SIZE$(I_L(t)) < \frac{4\omega h_B}{\epsilon}(h_B + 1)$ then        // Offline mode
    Use offline algorithm
else                                                                  // Online mode
    Find suitable group $g = (l, X, r)$ according to Definition 3.1
    Shift($g, \{i_t\}$)
    BlockBalancing
```

New containers. We already mentioned that most of the repacking happens inside existing containers and therefore new containers occur rather in special cases. However, note that these special cases are important: Items which have to be shifted out of group $(l, A, 0)$ can not be shifted further, as there is no group to the left (see Figures 2 and 4).

Therefore, we also have to deal with new containers in the container packing. Obviously, updating the level assignment such that new containers are placed in new levels is not enough to guarantee a good competitive ratio. Instead, a new level assignment has to be found, which maintains large parts of the existing assignment (in order to bound the repacking). Since this problem is closely related to an online bin packing problem, here we make use of the LP/ILP-techniques developed in [17]. For all technical details see [18].

3.2 Insertion Algorithm

Let $I_L(t) = \{i_1, i_2, \ldots, i_t\}$ denote the instance at time t. The insertion algorithm for big items, given in Algorithm 2, works in one of two modes: While SIZE$(I_L(t)) < \frac{4\omega h_B}{\epsilon}(h_B + 1)$, Algorithm 2 works in the *offline mode*. Here, an offline algorithm fulfilling all invariant properties repacks the whole instance each time a new item arrives. This is due to the fact that the operations modifying the LP-solutions require a minimum size of $I_L(t)$. As soon as SIZE$(I_L(t))$ is large enough, in the *online mode* the algorithm goes over to use Shift$(g, \{i_t\})$ to insert i_t into the suitable group g.

The last operation in Algorithm 2, denoted as BlockBalancing, adapts the total number of containers to the increasing value of SIZE$(I_L(t))$. Recall that by choice of the parameters (see Section 2.3), k depends on SIZE$(I_L(t))$ and thus increases over time. That is, at some point the parameter k changes to $k' = k + 1$. Obviously, we can not rebuild the whole container assignment to fulfill the new group sizes required by (c-d) according to the new parameter k'. Instead, the block structure (see Section 2.1) is exactly designed to deal with this situation: All groups of block A that satisfy invariant property (c) with parameter k satisfy (d) for parameter k', if they were in block B. In the procedure BlockBalancing groups are moved from block B to A parallel to the increasing fractional value of k. When block B is empty, groups from block A can be 'renamed' to block B groups. This way, (c-d) are fulfilled for the new parameter k' and the repacking is distributed among all time steps since the last parameter update. This technique was developed in [4]. For more details and a precise description of the operations see [18].

With respective results for Shift (including Lemma 3.2) and BlockBalancing, Algorithm 2 maintains all invariant properties. Furthermore, we can show that all operations modifying the LP/ILP-solutions of the level assignment return feasible solutions with the desired approximation guarantee. Therefore we obtain the following result:

▶ **Theorem 3.4.** *Algorithm 2 is an AFPTAS for the insertion of big items with asymptotic competitive ratio* $1 + \mathcal{O}(\epsilon)$.

4 Migration Analysis

It remains to analyze the migration factor of Algorithm 2. Recall the definition of the migration factor $\mu = \frac{\text{SIZE}(\mathit{Repacking}(t))}{\text{SIZE}(i_t)}$, where *Repacking*$(t)$ is the set of repacked items and i_t the item arriving at time t. Since in this extended abstract we focus on big items, the migration factor can be bound without amortization. First, note that in the offline mode of Algorithm 2, the repacking size is clearly bounded by $\text{SIZE}(I_L(t)) < \mathcal{O}\left(\frac{1}{\epsilon^5}\log\frac{1}{\epsilon}\right)$. The analysis for the online mode is quite involved since the operation SHIFT consists of several repacking steps performed in different groups. In the maximum shift sequence each group occurs once (see again Figure 2), thus the maximum number of shift operations can be at most the number of groups $|G|$. Again, one crucial argument is that $|G| \leq \mathcal{O}\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\right)$ (see Lemma 2.2). We give a detailed analysis for the repacking of the shift operation in [18] and get eventually:

▶ **Lemma 4.1.** *The total repacking in a maximum shift sequence is at most* $\mathcal{O}\left(\frac{1}{\epsilon^7}\left(\log\frac{1}{\epsilon}\right)^2\right)$.

Recall that in the online mode of Algorithm 2 the procedure BLOCKBALANCING performs repacking as well. However, it can be shown that its repacking size is dominated by the SHIFT part. Since big items have minimum size ϵ^2, we obtain the following corollary:

▶ **Corollary 4.2.** *Algorithm 2 has the migration factor* $\mu = \mathcal{O}\left(\frac{1}{\epsilon^9}\left(\log\frac{1}{\epsilon}\right)^2\right)$.

Acknowledgement. We would like to thank Marten Maack for helpful discussions on the lower bound (Section 1.3).

References

1 Brenda S. Baker, Edward G. Coffman, Jr, and Ronald L. Rivest. Orthogonal packings in two dimensions. *SIAM Journal on Computing*, 9(4):846–855, 1980.

2 Brenda S. Baker and Jerald S. Schwarz. Shelf algorithms for two-dimensional packing problems. *SIAM Journal on Computing*, 12(3):508–525, 1983.

3 János Balogh, József Békési, and Gábor Galambos. New lower bounds for certain classes of bin packing algorithms. *Theoretical Computer Science*, 440:1–13, 2012.

4 Sebastian Berndt, Klaus Jansen, and Kim-Manuel Klein. Fully dynamic bin packing revisited. In *International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, pages 135–151, 2015.

5 Donna J. Brown, Brenda S. Baker, and Howard P. Katseff. Lower bounds for on-line two-dimensional packing algorithms. *Acta Informatica*, 18(2):207–225, 1982.

6 Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 2017.

7 Edward G. Coffman, Jr, Michael R. Garey, David S. Johnson, and Robert E. Tarjan. Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM Journal on Computing*, 9(4):808–826, 1980.

8 János Csirik and Gerhard J. Woeginger. Shelf algorithms for on-line strip packing. *Information Processing Letters*, 63(4):171–175, 1997.

9 Kurt Eisemann. The trim problem. *Management Science*, 3(3):279–284, 1957.

10 Leah Epstein and Asaf Levin. A robust APTAS for the classical bin packing problem. *Mathematical Programming*, 119(1):33–49, 2009.

11 Giorgio Gambosi, Alberto Postiglione, and Maurizio Talamo. Algorithms for the relaxed online bin-packing model. *SIAM Journal on Computing*, 30(5):1532–1551, 2000.

12 Xin Han, Kazuo Iwama, Deshi Ye, and Guochuan Zhang. Strip packing vs. bin packing. In *International Conference on Algorithmic Applications in Management (AAIM)*, pages 358–367. Springer, 2007.

13 Rolf Harren, Klaus Jansen, Lars Prädel, and Rob Van Stee. A (5/3+ ε)-approximation for strip packing. *Computational Geometry*, 47(2):248–267, 2014.

14 Johann L. Hurink and Jacob J. Paulus. Online algorithm for parallel job scheduling and strip packing. In *International Workshop on Approximation and Online Algorithms (WAOA)*, pages 67–74. Springer, 2007.

15 Johann L. Hurink and Jacob J. Paulus. Online scheduling of parallel jobs on two machines is 2-competitive. *Operations Research Letters*, 36(1):51–56, 2008.

16 Csanád Imreh. Online strip packing with modifiable boxes. *Operations Research Letters*, 29(2):79–85, 2001.

17 Klaus Jansen and Kim-Manuel Klein. A robust AFPTAS for online bin packing with polynomial migration. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 589–600. Springer, 2013.

18 Klaus Jansen, Kim-Manuel Klein, Maria Kosche, and Leon Ladewig. Online strip packing with polynomial migration. *CoRR*, abs/1706.04939, 2017. URL: http://arxiv.org/abs/1706.04939.

19 Berit Johannes. Scheduling parallel jobs to minimize the makespan. *Journal of Scheduling*, 9(5):433–452, 2006.

20 Narendra Karmarkar and Richard M. Karp. An efficient approximation scheme for the one-dimensional bin-packing problem. In *Foundations of Computer Science (FOCS)*, pages 312–320, Nov 1982.

21 Claire Kenyon and Eric Rémila. A near-optimal solution to a two-dimensional cutting stock problem. *Mathematics of Operations Research*, 25(4):645–656, 2000.

22 Walter Kern and Jacob J. Paulus. A note on the lower bound for online strip packing. *Statistics and Computing*, 2009.

23 Kirk Pruhs, Jiri Sgall, and Eric Torng. Online scheduling. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, pages 15–1, 2004.

24 Peter Sanders, Naveen Sivadasan, and Martin Skutella. Online scheduling with bounded migration. *Mathematics of Operations Research*, 34(2):481–498, 2009.

25 Steven S. Seiden. On the online bin packing problem. *Journal of the ACM (JACM)*, 49(5):640–671, 2002.

26 Martin Skutella and José Verschae. Robust polynomial-time approximation schemes for parallel machine scheduling with job arrivals and departures. *Mathematics of Operations Research*, 41(3):991–1021, 2016.

27 Christoph Steiger, Herbert Walder, Marco Platzner, and Lothar Thiele. Online scheduling and placement of real-time tasks to partially reconfigurable devices. In *Real-Time Systems Symposium, 2003. RTSS 2003. 24th IEEE*, pages 224–225. IEEE, 2003.

28 Deshi Ye, Xin Han, and Guochuan Zhang. A note on online strip packing. *Journal of Combinatorial Optimization*, 17(4):417–423, 2009.

29 Deshi Ye, Xin Han, and Guochuan Zhang. Online multiple-strip packing. *Theoretical Computer Science*, 412(3):233–239, 2011.

30 Guosong Yu, Yanling Mao, and Jiaoliao Xiao. A new lower bound for online strip packing. *European Journal of Operational Research*, 250(3):754–759, 2016.

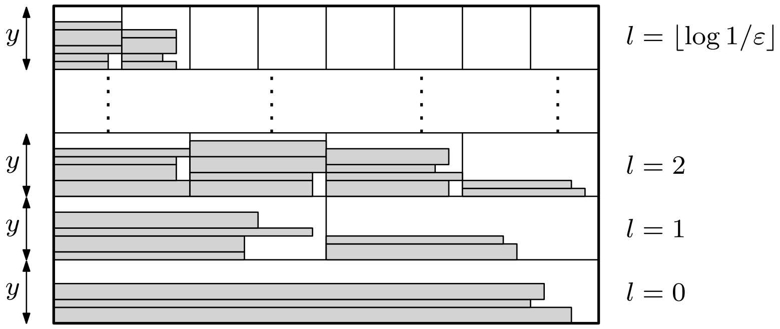


Figure 6 F-buffer contains 2^l slots of height y for each category l.

A Flat Items

We say an item i is *flat* if $w(i) \geq \epsilon$ and $h(i) < \epsilon$. The main difficulty of flat items becomes clear in the following scenario: Imagine that flat items of a group g are elements of $S_{out} = \text{WidestItems}(\cdot, \Delta)$ in a shifting process. Remember that generally each container, from which items are removed, has to be sinked (see Section 3.1), i. e. at most $|S_{out}|$ containers. In case of big items, due to their minimum height ϵ we get $|S_{out}| \leq \lfloor \Delta/\epsilon \rfloor$. In contrast, flat items can have an arbitrary small height and thus no such bound is possible. But Sink on all K_g containers would lead to unbounded migration (since K_g depends on $\text{SIZE}(I_L)$).

Therefore, we aim for a special packing structure for flat items that avoids the above problem of sinking too many containers. Like shown in Figure 1a, flat items build a sorted stack at the top of the container such that the least wide item is placed at the top edge. Thereby, widest items can be removed from the container without leaving a gap.

To maintain the sorting, we introduce a buffer for flat items called *F-buffer*. It is located in a rectangular segment of width 1 and height ωy, somewhere in the packing, for some constant y. Note that the additional height for the F-buffer is bounded by $\omega y = \mathcal{O}((\log 1/\epsilon) y)$. The internal structure of the F-buffer is shown in Figure 6: For each category l, there are 2^l slots in one level of height y. Items can be placed in any slot of their category.

An incoming flat item may overflow the F-buffer, more precisely, the level of one category in the F-buffer. For this purpose, Algorithm 3 iterates over all groups $g_q, g_{q-1}, \ldots, g_0$ of this category, where g_q is the rightmost and g_0 the leftmost group[4]. For each group, the set S contains those items in the F-buffer for which the group is suitable. The set S is split into smaller subsets of total height at most 1, then each subset gets inserted via a single call of Shift.

Note that the concept of a 'buffered insertion' for small items, like in Algorithm 3, corresponds with the notion of amortized migration: While flat items can be placed in the F-buffer, no repacking is performed at all. We save this migration for a later time step, namely when the F-buffer is full. Then, all items from the F-buffer get inserted into the containers, maintaining the packing structure and resulting in an empty F-buffer.

[4] Note that the direction of the iterative shifting is crucial: Calling Shift for a group g may reassign items in all groups left to g. Therefore, iterating from 'right to left' is necessary to guarantee that after shifting into group g, no group to the right of g is suitable for a remaining item in S. In other words, with this direction one shift call for each group is enough, which is in general not true for the direction 'left to right'.

```
Algorithm 3: Insertion of a flat item
  Input: Flat item $i_t$ of category $l$
if $i_t$ can be placed in the F-buffer then
    Place $i_t$ in the F-buffer
else
    Let $B(l)$ be the set of items in the buffer slots of category $l$
    for $j = q(l,A) + q(l,B) + 1, \dots, 0$ do
        Let $g_j = \begin{cases} (l,A,j) & j \le q(l,A) \\ (l,B,j-q(l,A)-1) & j > q(l,A) \end{cases}$
        Let $S = \{i \in B(l) \mid g_j \text{ is suitable for } i\}$
        Let $S_1, \dots, S_n$ be partition of $S$ with $h(S_r) \in (1-\epsilon, 1]$ for all $1 \le r \le n$.
        for $r = 1, \dots, n$ do
            Shift($g_j$,$S_r$)
        Remove $S$ from $B(l)$
    BlockBalancing
```

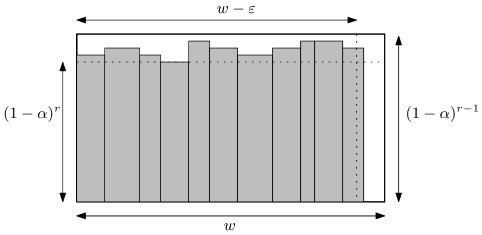


Figure 7 Shelf for narrow items of group r (dense).

B Narrow Items

We say an item i is *narrow* if $w(i) < \epsilon$. Narrow items can be packed efficiently if items of similar height are packed in a row. This is the concept of *shelf algorithms* introduced by Baker and Schwarz [2] which is described in the next subsection.

However, the goal is to integrate narrow items into the container packing introduced in Section 2. We show in Section B.2 how to fill gaps in the container packing with shelfs of narrow items. This leads to a modified first-fit-algorithm for narrow items with asymptotic approximation ratio of $1 + \mathcal{O}(\epsilon)$, as finally shown in Lemma B.2.

B.1 Shelf Packing

For a parameter $\alpha \in (0,1)$ item i belongs to *group* $r \in \mathbb{N} \setminus \{0\}$ if $h(i) \in [(1-\alpha)^r, (1-\alpha)^{r-1})$. Narrow items of group r are placed into a *shelf of group* r, which is a rectangle of height $(1-\alpha)^{r-1}$. Figure 7 shows a shelf for group r. Analogously to [2], we say a shelf of width w is *dense* when it contains items of total width greater than $w - \epsilon$ and *sparse* otherwise.

When the instance consists only of narrow items, the concept of shelf algorithms yields an online AFPTAS immediately. Consider the following first-fit shelf algorithm: Place an item of group r into the first shelf of group r where it fits, open a new shelf of group r only if necessary[5].

[5] Note that this simple algorithm works in the online setting since no sorting is necessary (in contrast to the NFDH algorithm [7], for example).

▶ **Lemma B.1.** *If I contains only narrow items, the shelf algorithm with parameter $\alpha = \frac{\epsilon^2}{1-\epsilon^2}$ yields a packing of height at most $(1+\epsilon)\,\mathrm{OPT}(I) + \mathcal{O}\left(1/\epsilon^4\right)$.*

Proof. For a group r, let $I_r = \{i \in I \mid h(i) \in [(1-\alpha)^r, (1-\alpha)^{r-1})\}$. Consider the packing obtained by the shelf algorithm and let β_r the number of shelfs of group r. Each dense shelf for group r contains items of size at least $(1-\alpha)^r(1-\epsilon)$, see Figure 7. Note that by the first-fit-principle, for each group at most one shelf is sparse. Thus there are at least $\beta_r - 1$ dense shelfs for each group r, hence $\mathrm{SIZE}(I_r) \geq (\beta_r - 1)(1-\alpha)^r(1-\epsilon)$, or equivalently

$$\beta_r \leq \mathrm{SIZE}(I_r)(1-\alpha)^{-r}(1-\epsilon)^{-1} + 1\,. \tag{3}$$

The packing consists of β_r shelfs of height $(1-\alpha)^{r-1}$ for each group r (set $\beta_r = 0$, if the group does not exist). Therefore, the packing height is:

$$\begin{aligned}
&\sum_{r=0}^{\infty} \beta_r (1-\alpha)^{r-1} \\
&\leq \sum_{r=0}^{\infty} \left(\mathrm{SIZE}(I_r)(1-\alpha)^{-r}(1-\epsilon)^{-1} + 1\right)(1-\alpha)^{r-1} && \text{eq. (3)} \\
&= \sum_{r=0}^{\infty} \mathrm{SIZE}(I_r)(1-\alpha)^{-1}(1-\epsilon)^{-1} + (1-\alpha)^{r-1} \\
&= \frac{1}{(1-\epsilon)(1-\alpha)} \sum_{r=0}^{\infty} \mathrm{SIZE}(I_r) + \sum_{r=0}^{\infty}(1-\alpha)^{r-1} \\
&\leq \frac{1}{(1-\epsilon)(1-\alpha)} \mathrm{SIZE}(I) + \frac{1}{\alpha - \alpha^2} && \text{Geometric series} \\
&\leq \frac{1}{(1-\epsilon)(1-\alpha)} \mathrm{OPT}(I) + \frac{1}{\alpha - \alpha^2} \\
&= (1+\epsilon)\,\mathrm{OPT}(I) + \mathcal{O}\left(\frac{1}{\epsilon^4}\right) && \text{Choice of } \alpha
\end{aligned}$$

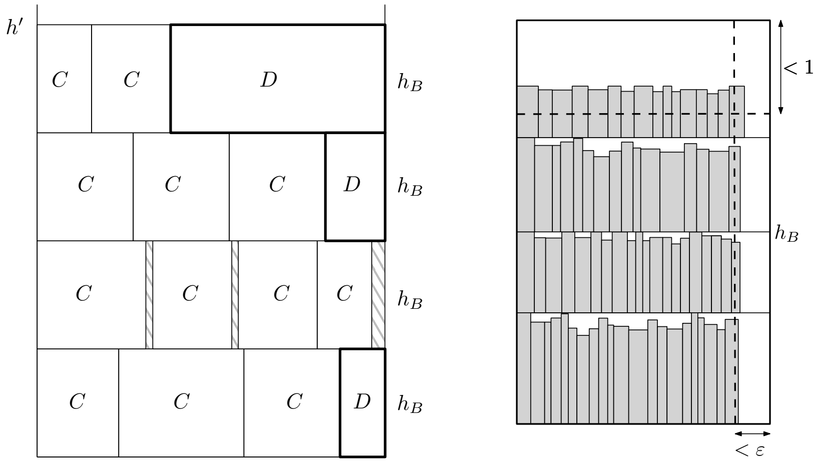
Note that the total height of sparse shelfs is bounded by a constant, even if the number of groups is unbounded. This follows by the geometric series:

$$\sum_{r=0}^{\infty}(1-\alpha)^{r-1} = \frac{1}{1-\alpha}\sum_{r=0}^{\infty}(1-\alpha)^r = \frac{1}{1-\alpha}\frac{1}{\alpha} = \frac{1}{\alpha-\alpha^2}$$ ◀

B.2 Filling Gaps in the Container Packing

As shown in the previous section, shelfs are a good way to pack narrow items efficiently. But before opening a new shelf that increases the packing height, we have to ensure that the existing packing is well-filled. Therefore, the idea is to fill gaps in the container packing with shelfs of narrow items first. Thereby, a *gap* is the rectangle of height h_B that fills the remaining width of a level. Only if no significant gaps exist, new shelfs are packed on top of the packing.

Figure 8a shows the packing structure of the strip on a high level: Here, all C-rectangles represent containers for big and flat items. If the total width of containers in a level is less than a threshold value (say $1 - \mathcal{O}(\epsilon)$), these containers get *aligned* such that the only gap occurs at the right end of a level. We call these gaps D-containers. Inside, each D-container is organized in shelfs of narrow items (see Figure 8b).



(a) Well-filled container packing **(b)** D-container

Figure 8 D-containers are introduced to fill gaps in the container packing with shelfs.

Since the width of containers changes due to shift operations, without aligning a level could be fragmented such that no contiguous area can be used for a D-container. As aligning levels means further repacking, the insertion algorithm for narrow items makes use of a special buffer, similar to the case of flat items.

Finally, it has to be proven that inserting narrow items this way maintains the overall approximation guarantee. Note that the first-fit strategy for narrow items (sketched above), has two important properties: If the item can be placed in a gap, the packing height does not increase. Now suppose that an item gets placed in a new shelf on top of the packing. This only occurs, if the existing packing is well-filled, since no significant gaps were found. As a consequence of this important observation we get the following lemma (proven in [18]):

▶ **Lemma B.2.** *Let h' be the height of the container packing. The insertion of narrow items returns a packing of height h_{final}, such that $h_{final} \leq \max\left\{h', (1+\epsilon')\,\mathrm{SIZE}(I) + \mathcal{O}\left(\frac{\omega}{\epsilon^3}\right)\right\}$, where $\epsilon' \in \mathcal{O}(\epsilon)$.*

Note that I denotes the set of all items (including big, flat, and narrow items). Lemma B.2 immediately implies that the final packing height is at most $(1+\mathcal{O}(\epsilon))\,\mathrm{OPT}(I)+\mathcal{O}(poly(1/\epsilon))$: We can use Lemma 2.3 to bound the height h' of the container packing and the fact that $\mathrm{SIZE}(I) \leq \mathrm{OPT}(I)$.

Density Independent Algorithms for Sparsifying k-Step Random Walks*

Gorav Jindal[1], Pavel Kolev[2], Richard Peng[3], and Saurabh Sawlani[4]

1 **Max-Planck-Institut für Informatik, Saarbrücken, Germany**
gjindal@mpi-inf.mpg.de

2 **Max-Planck-Institut für Informatik, Saarbrücken, Germany**
pkolev@mpi-inf.mpg.de

3 **Georgia Institute of Technology, Atlanta, GA, USA**
rpeng@gatech.edu

4 **Georgia Institute of Technology, Atlanta, GA, USA**
sawlani@gatech.edu

Abstract

We give faster algorithms for producing sparse approximations of the transition matrices of k-step random walks on undirected and weighted graphs. These transition matrices also form graphs, and arise as intermediate objects in a variety of graph algorithms. Our improvements are based on a better understanding of processes that sample such walks, as well as tighter bounds on key weights underlying these sampling processes. On a graph with n vertices and m edges, our algorithm produces a graph with about $n \log n$ edges that approximates the k-step random walk graph in about $m + k^2 n \log^4 n$ time. In order to obtain this runtime bound, we also revisit "density independent" algorithms for sparsifying graphs whose runtime overhead is expressed only in terms of the number of vertices.

1998 ACM Subject Classification F.2.2. [Nonnumerical Algorithms and Problems] Computations on Discrete Structures, G.3 [Probability and Statistics] Probabilistic algorithms (including Monte Carlo)

Keywords and phrases random walks, graph sparsification, spectral graph theory, effective resistances

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.14

1 Introduction

Random walks are some of the most natural mathematical objects, and have historically been used to model processes in fields ranging from psychology to economics. Problems related to random walks on graphs, such as shortest path and minimum cut are well studied in both static [34] and dynamic settings [21, 17]. While some of these problems, such as shortest path, aim to find a single walk, other problems such as finding flows/cuts [16] or triangle densities [4, 38] aim to capture information related to collections of walks. Algorithms and data structures for such problems often need to store, or can be sped up by, intermediate structures that capture the global properties of multi-step walks [31, 18, 1, 3]. However, many intermediate structures are inherently dense and therefore expensive to compute explicitly.

* Pavel Kolev is funded by the Cluster of Excellence "Multimodal Computing and Interaction" within the Excellence Initiative of the German Federal Government. Richard Peng and Saurabh Sawlani are partially supported by the NSF under Grant No. 1637566.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 14; pp. 14:1–14:17
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Graph sparsification is a technique for efficiently approximating a dense graph by a sparser one, while preserving some key properties such as sizes of graph cuts, distances between vertices, or linear operator properties of matrices associated with the graphs. Spectral sparsifiers are the ones which guarantee linear operator approximations, but they also inherently approximate all graph cuts. Moreover, they have various applications in graph algorithms, such as sampling from graphical models [7], solving linear systems [32, 27], sampling random spanning trees [13, 14][1] and maintaining approximate minimum cuts in dynamically changing graphs [1]. In these applications, the optimal performance is achieved by producing a sparsifier of a denser intermediate object directly, instead of generating the exact larger object. Of these intermediate objects, some of the more commonly studied are random walk matrices [32, 7]. These matrices contain the pairwise transition probabilities between vertices under k-step walks. Moreover, such matrices are dense even for sparse original graphs with k as small as 2: for instance, the 2-step walk on the n-vertex star contains an $n-1$ sized clique.

Cheng et al. [7] studied random walk sparsification, and gave a routine that produces an ϵ-spectral sparsifier (which we will formally define in Subsection 2.2) with $O(\epsilon^{-2} n \log n)$ edges for a k-step walk matrix in $O(\epsilon^{-2} k^2 m \log^{O(1)} n)$ time. Our main result, which we show in Section 3, is a direct improvement of that routine:

▶ **Theorem 1** (Sparsifying Laplacian Monomials). *Given a graph G and an error $\varepsilon \in (0,1)$, there is an algorithm that outputs an ε-spectral sparsifier of G^k with at most $O(\varepsilon^{-2} n \log n)$ edges in $\widehat{O}(m + k^2 \varepsilon^{-2} n \log^4 n)$ time.*[2]

We term this type of running time with most of the overhead on the number of vertices, n, as density independent. Such runtimes arise naturally in many other graph problems [15], and was first studied for graph sparsification in an earlier manuscript by a subset of the authors [22], where the authors sparsify certain Laplacian monomials (specifically, monomials where the degree is a power of 2) in $O(m \log^2 n + \epsilon^{-4} n \log^4 n \log^5 k)$ time. They also extend this to specific classes of matrix polynomials - those with coefficients induced by "mixture of discrete Binomial distributions" with similar running-times. Our algorithm can also be combined with the repeated-squaring technique in [7] to reduce the runtime dependence on k to logarithmic [8]. Additionally, if we generalize our results from monomials to general random walk polynomials [8], this would then supersede all claims from [22]. As these steps are much closer to the ideas in [7], we will focus on the small k case in this paper. Furthermore, as our sparsification algorithm has a much more direct interaction with routines that provide upper bounds of effective resistances, they can likely be combined with tools from [1] to give dynamic algorithms for maintaining G^k under insertions/deletions to G. However, as there are currently only few applications of such sparsifiers, we believe it may be more fruitful to extend the applications before further developing the tools.

Our algorithms, as with the ones from [22, 7] are based on implicit sampling of dense graphs by probabilities related to effective resistances. Our improvements rely on an a key insight from the sparse Gaussian elimination algorithm by Kyng and Sachdeva [29]: using triangle inequality between effective resistances to obtain a tighter set of probability

[1] While these manuscripts are simultaneous, the significantly earlier original proposal of density independent sparsification of walks [22], and the importance of it in the algorithm of [14] were major motivations for this paper.

[2] We use $\widehat{O}$ to denote the omission of logarithmic terms lower than the ones shown in the set. In all cases in this paper, we track terms of $\log n$ explicitly and such notation hides terms of $\log\log n$. In all these cases, this notation hides a term of at most $(\log\log n)^2$.

upper bounds. So, to sample an edge in G^k, we essentially simulate a k-step walk in G by first sampling an edge, and then "walking" along both directions to make a length k walk. A simple but crucial detail in the algorithm is the selection of that first edge. Instead of sampling it uniformly as in [7], we pick an edge e with probability proportional to the product of its weight and effective resistance (its "leverage score"). Although the change is subtle, this helps remove any sampling count dependencies on the number of edges, making a density-independent runtime possible.

Obtaining density-independent bounds is critical for graph sparsification algorithms, since they are primarily invoked on relatively dense graphs. A graph sparsification routine that produces a sparsifier with $\widehat{O}(n \log^2 n)$ edges in $\widehat{O}(m \log^2 n)$ time, such as the combinatorial algorithm given in [28], will only be invoked when $m > n \log^2 n$, which means that the running time of the algorithm is actually $\Omega(n \log^4 n)$. Additionally, a desired property of a sparsification algorithm is that applying it repeatedly does not cause a blow up in its running time. One way to achieve this is to ensure that the running time is linear in the number of edges, and the overhead is only on the number of vertices. As a result, we believe that for graph sparsification to work as a primitive for processing large graphs, a running time of $\widehat{O}(m + n \log^2 n)$, or better, is necessary.

In Section 4, we provide some steps toward this direction by outlining a better density-independent spectral sparsification algorithm. We combine ideas from previous density-independent algorithms for sparsifying graphs [26] with recent developments in tree embedding and numerical algorithms to obtain numerical sparsification routines that run in $\widehat{O}(m + n \log^4 n)$ time, and combinatorial ones that take $\widehat{O}(m + n \log^6 n)$ time. Although these routines do not involve new ideas, they utilize some of the latest machinery, and give the current best time-bounds for density-independent sparsification. Importantly, both of these routines are in turn applicable to the walk sparsification algorithm in Section 3, giving routines for sparsifying k-step walks with similar running times: the bound stated in Theorem 1 is via the numerical routine. While these results are far from what we think are the best possible, we show a variety of new algorithmic tools for designing algorithms that sparsify k-step random walks matrices.

Our methods of extending density independent sparsification to random walks play a crucial role in several other types of graph sparsification - in particular, sparsification routines requiring only an oracle that samples edges from a distribution of approximate resistances, and oracle access to approximate leverage scores. Even for the 'simpler' problem of producing cut sparsifiers of G^k, these are the only known efficient approaches.

For instance, the routines for approximately sampling and counting spanning trees from [14] rely on producing determinant preserving sparsifiers of Schur complements of graph Laplacians, which are themselves sums of random walks. Specifically, the algorithm in [14] builds a sparse graph H such that $(1 - \epsilon) \det(G) \leq \det(H) \leq (1 + \epsilon) \det(G)$, but requires an overhead of about $\Theta(\sqrt{n})$ samples, leading to sparsifiers with about $\Theta(\epsilon^{-2} n^{1.5})$ edges. On the other hand, the number of calls this algorithm makes to the oracles is given by $O(n^{-1} \sum_{e \in E} \ell_e)$, where ℓ_e is a value dominating the leverage score of e. Thus, to extend their algorithm to Schur complements and simultaneously guarantee that the time-bound does not blow up beyond $O(n^{1.5})$, we have to ensure that these approximate leverages scores still sum up to $O(n)$. This is similar to our requirement, and is done by picking the initial edge of the random walk with probability proportional to the product of its weight and its effective resistance, and extending it to a walk on both sides.

2 Background

We start with some background information about graphs and matrices corresponding to them. These matrices allow us to define graph approximations, as well as compute key sampling probabilities needed to produce spectral sparsifiers. Due to space constraints, we will only formally define most of the concepts. More intuition on them can be found in notes on spectral graph theory and random walks such as [12, 30].

2.1 Random Walks and Matrices

Let $G = (V, E, \boldsymbol{w})$ be a weighted undirected graph. We define its adjacency matrix $\boldsymbol{A}$ as $\boldsymbol{A}_{uv} \overset{\text{def}}{=} \boldsymbol{w}_{uv}$, and its degree matrix $\boldsymbol{D}$ as $\boldsymbol{D}_{uu} \overset{\text{def}}{=} \sum_{v \in V} w_{uv}$ and $\boldsymbol{D}_{uv} \overset{\text{def}}{=} 0$ when $u \neq v$. This leads to the graph Laplacian $\boldsymbol{L}_G \overset{\text{def}}{=} \boldsymbol{D} - \boldsymbol{A}$.

One step of a random walk can be viewed as distributing the 'probability mass' at a vertex evenly among the edges leaving it, and passing them onto its neighbors. In terms of these matrices, it is equivalent to first dividing by $\boldsymbol{D}$, and then multiplying by $\boldsymbol{A}$. Thus, the left transition matrix of the k^{th} step random walk is given by $(\boldsymbol{D}^{-1}\boldsymbol{A})^k$. The corresponding Laplacian matrix of the k-step random walk is defined by

$$\boldsymbol{L}_{G^k} \overset{\text{def}}{=} \boldsymbol{D} - \boldsymbol{A}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{k-1}.$$

The matrices $\boldsymbol{A}(\boldsymbol{D}^{-1}\boldsymbol{A})^{k-1}$ can be viewed as a sum over length k walks. This view is particularly useful in our algorithm, as well as the earlier walk sparsification algorithm by Cheng et al. [7] because these walks are a more 'natural' unit upon which sparsification by effective resistances is applied. Formally, we can define the weight of a length k walk $(u_0, u_1, \ldots, u_k)$ by

$$\boldsymbol{w}_{(u_0,u_1,\ldots,u_k)} \overset{\text{def}}{=} \frac{\prod_{i=1}^{k} \boldsymbol{w}_{u_{i-1},u_i}}{\prod_{i=1}^{k-1} \boldsymbol{d}_{u_i}}. \tag{1}$$

Straightforward checking shows that for any $u_0, u_k \in V$, the weight of the edge (u_0, u_k) in $\boldsymbol{G}^k$ is given by

$$\boldsymbol{w}^{\boldsymbol{G}^k}_{u_0,u_k} \overset{\text{def}}{=} \left[\boldsymbol{A}\left(\boldsymbol{D}^{-1}\boldsymbol{A}\right)^{k-1}\right]_{u_0u_k} = \sum_{u_1,\ldots,u_{k-1}} \boldsymbol{w}_{(u_0,u_1,\ldots,u_k)}. \tag{2}$$

2.2 Spectral Approximations of Graphs

Our notion of matrix approximations will be through the $\approx$ symbol, which is in turn defined through the Löewner partial ordering of matrices. For two matrices, $\boldsymbol{A}$, and $\boldsymbol{B}$, we say that $\boldsymbol{A} \preceq \boldsymbol{B}$ if $\boldsymbol{B} - \boldsymbol{A}$ is positive semidefinite, and $\boldsymbol{A} \approx_\kappa \boldsymbol{B}$ if there exists bounds $\lambda_{\min}$ and $\lambda_{\max}$ such that $\lambda_{\min}\boldsymbol{A} \preceq \boldsymbol{B} \preceq \lambda_{\max}\boldsymbol{A}$, and $\lambda_{\max} \leq \kappa\lambda_{\min}$. This notation is identical to generalized eigenvalues, and in particular, $\boldsymbol{L}_G \approx_\kappa \boldsymbol{L}_H$ implies that all cuts on them are within a factor of κ of each other.

The adjacency matrix of a graph has several undesirable properties when it comes to operator based approximations: it can have a large number of eigenvalues at 0, which must be exactly preserved under relative error approximations. As a result, graph approximations are defined in terms of graph Laplacians. As we will discuss below, these approximations are often in terms of reducing edges. So formally, we say that a graph H is a κ-sparsifier of G if $\boldsymbol{L}_H \approx_\kappa \boldsymbol{L}_G$, and our goal is to compute an ϵ-sparsifier of the k-step random walk graph G^k.

Algorithm 1 IdealSample$(G, \varepsilon, \widetilde{\boldsymbol{\tau}})$

Input: A graph $G = (V, E, \boldsymbol{w})$, an integer k, and leverage score upper bounds $\widetilde{\boldsymbol{\tau}}_e$ that satisfy $\widetilde{\boldsymbol{\tau}}_e \geq \boldsymbol{w}_e \mathcal{R}^G_{\text{eff}}(e)$ for all edges e.
Output: An ε-sparsifier $\boldsymbol{H}$ of G with $O(\varepsilon^{-2}\mathcal{T}\log n)$ edges, where $\mathcal{T} = \sum_{e\in E}\widetilde{\boldsymbol{\tau}}_e$.

```
1. Initiate $\boldsymbol{H}$ as an empty graph.
2. Set sample count $N \leftarrow O(\varepsilon^{-2}\mathcal{T}\log n)$.
3. Repeat $N$ times:
   a. Pick an edge $e$ in $G$ with probability $p_e = \widetilde{\boldsymbol{\tau}}_e/\mathcal{T}$.
   b. Add $e$ to $\boldsymbol{H}$ with new weight $\boldsymbol{w}_e/(Np_e)$.
```

2.3 Graph Sparsification by Effective Resistances

There are two ways of viewing graph sparsification: either as tossing coins independently on the edges, or sampling a number of them from an overall probability distribution. We take the second view here because it is expensive to access all edges in G^k. The pseudocode of the generic sampling scheme is given in Algorithm 1.

Algorithmically, the sampling step can be implemented by first generating a number uniformly random in $[0, \sum_e \widetilde{\boldsymbol{\tau}}_e]$, (considering we want an edge e to be chosen with probability proportional to a real number $\widetilde{\boldsymbol{\tau}}_e$) and binary searching among the prefix sums of the $\widetilde{\boldsymbol{\tau}}_e$ values until it reaches the edge corresponding to that point. In the RealRAM model [5, 33] of computation, however, this can be done in $O(m)$ preprocessing time and $O(1)$ query time using "pairing" or "aliasing" [6, 24, 39].

The guarantees of this routine require defining effective resistances and leverage scores. Effective resistance is a metric on a graph that is defined by:

$$\mathcal{R}^G_{\text{eff}}(u, v) \stackrel{\text{def}}{=} \boldsymbol{\chi}_{uv}^T \boldsymbol{L}_G^{\dagger} \boldsymbol{\chi}_{uv}, \tag{3}$$

where $\boldsymbol{L}_G^{\dagger}$ denotes the pseudoinverse of $\boldsymbol{L}_G$ and $\boldsymbol{\chi}_{uv}$ is the indicator vector with 1 at u and -1 at v. Intuitively, viewing the graph as an electrical network where an edge e acts as a resistor having resistance $1/\boldsymbol{w}_e$, the effective resistance between u and v is the potential difference required between them so that one unit of current flows from u to v.

The effective resistances $\mathcal{R}^G_{\text{eff}}$ are directly related to the statistical leverage scores $\boldsymbol{\tau}$ by the relation $\boldsymbol{\tau}_e = \boldsymbol{w}_e \mathcal{R}^G_{\text{eff}}(e)$. Moreover, these scores are well defined for general matrices, and have a wide range of applications in randomized linear algebra [40, 9, 11]. The guarantees of sampling by weight times effective resistance, or leverage scores, can then be formalized as:

▶ **Lemma 2.** *(Sampling by Upper Bounds on Leverage Scores [37]) Suppose $G = (V, E, w)$ is a graph and $\widetilde{\boldsymbol{\tau}}$ is a vector such that $\widetilde{\boldsymbol{\tau}}_e \geq \boldsymbol{w}_e \mathcal{R}^G_{\text{eff}}(e)$ for every edge e, then, with high probability, any process that simulates the ideal sampling in Algorithm 1 produces an ε-sparsifier of G with $O(\varepsilon^{-2}\mathcal{T}\log n)$ edges in $O(m + \varepsilon^{-2}\mathcal{T}\log^2 n)$ time, where $\mathcal{T} = \sum_{e\in E}\widetilde{\boldsymbol{\tau}}_e$.*

Proof Sketch. A variant (in page 10 of [20]) of the Matrix Chernoff bound [37] states that if $Y = \sum_{i=1}^N Y_i$, $Z = E[Y]$ and $0 \preceq Y_i \preceq RZ$ for every $i \in [k]$ and some scalar R, then for any $\epsilon \in (0, 1)$, it holds

$$Pr\left[(1-\epsilon)Z \preceq Y \preceq (1+\epsilon)Z\right] \geq 1 - 2n \cdot \exp\left\{\frac{-\epsilon^2}{3R}\right\}.$$

Setting Y_i to be the Laplacian of the scaled ith edge added to H in Step 3 of Algorithm 1, we have $Y_i = \dfrac{\boldsymbol{w}_e}{\widetilde{\boldsymbol{\tau}}_e \cdot O(\varepsilon^{-2}\log n)}\chi_e\chi_e^T$ and $Y = \boldsymbol{L}_{\boldsymbol{H}}$. Moreover, $E[\boldsymbol{H}] = G$ and thus $Z = \boldsymbol{L}_G$.

To prove that H is almost always an ϵ-sparsifier of G, it suffices to show that there is a scalar R such that $Y_i \preceq RZ$ for small enough R. Since $\boldsymbol{w}_e \chi_e \chi_e^T \preceq \boldsymbol{\tau}_e \boldsymbol{L}_G$ for every edge e (cf. [9, equation (11) in the proof of Lemma 11]) and by assumption $\boldsymbol{\tau}_e \leq \widetilde{\boldsymbol{\tau}}_e$, it follows that $Y_i \preceq R\boldsymbol{L}_G$ for $R = \Theta(\varepsilon^2/\log n)$. Hence, the desired bound on the failure probability holds.

The runtime follows by noting that in $O(m)$ time we can precompute prefix sums of $\widetilde{\boldsymbol{\tau}}$ and each consecutive edge sample takes $O(\log n)$ time using binary search. ◀

The bound on the number of samples then follows by:

▶ **Fact 3** (Foster's Theorem). *For any undirected graph $G = (V, E, \boldsymbol{w})$, it holds that*

$$\sum_{e \in E} \boldsymbol{w}_e \mathcal{R}_{\text{eff}}^{G}(u, v) = n - 1.$$

Leverage scores are the preferred objects for defining sampling distributions as they are scale invariant: doubling the weights of all edges does not change their leverages scores. However, we will still make extensive uses of effective resistances because of the need to approximate them across different graphs. Such approximations are difficult to state for leverage scores because spectrally similar graphs may have very different sets of combinatorial edges.

▶ **Fact 4.** *If G and H are graphs such that $\boldsymbol{L}_G \preceq \boldsymbol{L}_H$, then for any vertices u and v we have*

$$\mathcal{R}_{\text{eff}}^{H}(u, v) \leq \mathcal{R}_{\text{eff}}^{G}(u, v).$$

Note that this generalizes Rayleigh's monotonicity law, which postulates that the effective resistances can only increase as one removes edges from a graph.

3 Random Walk Sparsification via Walk Sampling

In this section, we describe our improved algorithm for sparsifying random walk polynomials. The main difficulty we need to overcome here is that the actual random walk matrix cannot be constructed explicitly. Instead, we need to simulate the ideal sampling routine shown in Algorithm 1 by constructing nearly tight upper bounds of effective resistances in G^k that can also be efficiently sampled from, without having explicit access to G^k.

To obtain these effective resistances estimates in G^k, the following lemma from [7] provides a helpful starting point.

▶ **Lemma 5.** *[7] For odd k, we have $\frac{1}{2}\boldsymbol{L}_G \preceq \boldsymbol{L}_{G^k} \preceq k\boldsymbol{L}_G$ and for even k, we have $\boldsymbol{L}_{G^2} \preceq \boldsymbol{L}_{G^k} \preceq \frac{k}{2}\boldsymbol{L}_{G^2}$.*

Furthermore, note that Lemma 5 combined with Fact 4 implies for odd k that

$$\mathcal{R}_{\text{eff}}^{G^k}(u, v) \leq 2\mathcal{R}_{\text{eff}}^{G}(u, v) \tag{4}$$

and for even k that

$$\mathcal{R}_{\text{eff}}^{G^k}(u, v) \leq \mathcal{R}_{\text{eff}}^{G^2}(u, v). \tag{5}$$

Since G^k might be dense, i.e. $E[G^k] = \Theta(n^2)$, it is prohibitive to use (4) and (5) directly. Instead, we upper bound the values with a random walk using the triangle inequality of effective resistances [36, Lemma 9.6.1].

▶ **Fact 6** (Triangle Inequality for Effective Resistances). *For any graph G and any walk $(u_0, u_1, \ldots, u_k)$, we have*

$$\mathcal{R}_{\text{eff}}^{G}(u_0, u_k) \leq \sum_{0 \leq i < k} \mathcal{R}_{\text{eff}}^{G}(u_i, u_{i+1}). \tag{6}$$

Now, suppose we have a vector $\widetilde{\boldsymbol{r}}$ that upper bounds the effective resistances, i.e., $\widetilde{\boldsymbol{r}}_e \geq \mathcal{R}_{\text{eff}}^{G}(e)$ for all e. Then, by Lemma 2 and Fact 6, to sparsify G^k, it suffices to sample a length k random walk in G with probability proportional to

$$\boldsymbol{w}_{(u_0,u_1,\ldots,u_k)} \cdot \sum_{0 \leq i < k} \widetilde{\boldsymbol{r}}_{u_i,u_{i+1}}. \tag{7}$$

This distribution has the advantage that it is efficiently computable:

▶ **Lemma 7.** *For any graph $G = (V, E, \boldsymbol{w})$, and any vector $\widetilde{\boldsymbol{r}} \in \mathbb{R}^E$, we can sample length k walks such that the probability of sampling the walk $(u_0, u_1, \ldots, u_k)$ is proportional to*

$$\boldsymbol{w}_{(u_0,u_1,\ldots,u_k)} \cdot \sum_{i=0}^{k-1} \widetilde{\boldsymbol{r}}_{u_i,u_{i+1}}$$

using the following procedure:

1. *Pick uniformly at random an index i in the range $[0, k-1]$.*
2. *Choose an edge (u_i, u_{i+1}) with probability proportional to $\boldsymbol{w}_e \widetilde{\boldsymbol{r}}_e$.*
3. *Extend the walk in both directions from u_i and u_{i+1} via two random walks.*

Proof.

(Step 1) Let i be the selected number. The probability of this event is $1/k$.

(Step 2) The probability of selecting an edge (u_i, u_{i+1}) is $\dfrac{\boldsymbol{w}_{u_i,u_{i+1}} \widetilde{\boldsymbol{r}}_{u_i,u_{i+1}}}{\langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle}$.

(Step 3) Conditioned on the event that edge (u_i, u_{i+1}) is selected, the probability to sample a walk $(u_0, \ldots, u_k)$ equals

$$\left(\prod_{j=1}^{i} \frac{\boldsymbol{w}_{u_{j-1},u_j}}{\boldsymbol{d}_{u_j}} \right) \cdot \left(\prod_{j=i+1}^{k-1} \frac{\boldsymbol{w}_{u_j,u_{j+1}}}{\boldsymbol{d}_{u_j}} \right) = \frac{\boldsymbol{w}_{(u_0,u_1,\ldots,u_k)}}{\boldsymbol{w}_{u_i,u_{i+1}}}.$$

Thus, summing over all choices of i, and by the total law of probability, the probability of sampling the walk $(u_0, u_1, \ldots, u_k)$ is

$$\sum_{i=0}^{k-1} \frac{1}{k} \cdot \frac{\boldsymbol{w}_{u_i,u_{i+1}} \widetilde{\boldsymbol{r}}_{u_i,u_{i+1}}}{\langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle} \cdot \frac{\boldsymbol{w}_{(u_0,u_1,\ldots,u_k)}}{\boldsymbol{w}_{u_i,u_{i+1}}} = \frac{\boldsymbol{w}_{(u_0,u_1,\ldots,u_k)}}{k \langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle} \sum_{i=0}^{k-1} \widetilde{\boldsymbol{r}}_{u_i,u_{i+1}}. \qquad \blacktriangleleft$$

The total number of samples needed by Lemma 2 is given by the summation over all length k random walks, similarly to [7, Lemma 29]. For completeness, we present its proof in Appendix A.

▶ **Lemma 8.** *For any weighted graph $G = (V, E, \boldsymbol{w})$, any $k \in \mathbb{N}_+$, and any vector $\widetilde{\boldsymbol{r}} \in \mathbb{R}^E$, it holds that*

$$\sum_{(u_0,u_1,\ldots,u_k)} \boldsymbol{w}_{(u_0,u_1,\ldots,u_k)} \cdot \sum_{0 \leq i < k} \widetilde{\boldsymbol{r}}_{u_i,u_{i+1}} = k \cdot \langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle. \tag{8}$$

For every odd k, by setting $\widetilde{\boldsymbol{r}}$ to (an approximation of) $\mathcal{R}_{\text{eff}}^{G}$, yields an efficient sampling procedure due to (8) and Lemma 7.

However, when k is even, Lemma 5 gives a bound in terms of $\mathcal{R}_{\text{eff}}^{G^2}$ (not $\mathcal{R}_{\text{eff}}^{G}$), i.e. $\mathcal{R}_{\text{eff}}^{G^k}(u,v) \leq \mathcal{R}_{\text{eff}}^{G^2}(u,v)$. Hence, the distribution in Lemma 7 requires an access to the 2-step random walk matrix G^2, which might also be dense and therefore expensive to compute.

Moreover, suppose G is a 2-length path graph $u - v - w$, then $\mathcal{R}_{\text{eff}}^{G^2}(u,v) = +\infty$, since G^2 has only one edge (u,w) (and self-loops). A naive approach to tackle these issues is to substitute $\mathcal{R}_{\text{eff}}^{G^2}$ with $\mathcal{R}_{\text{eff}}^{G}$. However, this fails shortly since it is not true in general that

$$\mathcal{R}_{\text{eff}}^{G}(u,v) + \mathcal{R}_{\text{eff}}^{G}(v,w) \geq \mathcal{R}_{\text{eff}}^{G^2}(u,w). \tag{9}$$

In particular, (9) does not hold for the length 2 path example from above. To verify this, note that $\mathcal{R}_{\text{eff}}^{G}(u,u) + \mathcal{R}_{\text{eff}}^{G}(u,v)$ is a finite number, whereas $\mathcal{R}_{\text{eff}}^{G^2}(u,w) = +\infty$ since u and v are disconnected in G^2. For a non-degenerate example, let G be a triangle graph on vertices u, v, w with $\boldsymbol{w}_{uv} = \boldsymbol{w}_{vw} = 1$ and $\boldsymbol{w}_{uw} = 100$. Then, $\mathcal{R}_{\text{eff}}^{G}(u,u) + \mathcal{R}_{\text{eff}}^{G}(u,v) \approx 1$ and $\mathcal{R}_{\text{eff}}^{G^2}(u,v) \approx 50$.

We overcome this issue by using effective resistances from the "double cover" of G, instead. The "double cover" $G \times P_2$ is the tensor product of G and a path of length 1. Combinatorially, $G \times P_2$ is a bipartite graph with vertex sets $V^{(A)}, V^{(B)}$ each a copy of V such that for every edge $(u,v) \in G$ we insert in $G \times P_2$ the following two edges: $u^{(A)}v^{(B)}$ and $u^{(B)}v^{(A)}$ with $\boldsymbol{w}_{u^{(A)}v^{(B)}} = \boldsymbol{w}_{u^{(B)}v^{(A)}} = \boldsymbol{w}_{uv}$. The next lemma (proved in Appendix A) relates the effective resistances of G^2 and $G \times P_2$.

▶ **Lemma 9.** *For any vertices u and v in G, it holds*

$$\mathcal{R}_{\text{eff}}^{G^2}(u,v) = \mathcal{R}_{\text{eff}}^{G\times P_2}(u^{(A)}, v^{(A)}),$$

where $u^{(A)}$ and $v^{(A)}$ are the corresponding copies of u and v in $V^{(A)}$, respectively.

Lemma 9 combined with Fact 6, fixes (9) by upper bounding the effective resistance $\mathcal{R}_{\text{eff}}^{G^2}(\cdot)$ with summation of terms $\mathcal{R}_{\text{eff}}^{G\times P_2}(\cdot)$, i.e. for every edge (u,w) in G^2 it holds that

$$\mathcal{R}_{\text{eff}}^{G^2}(u,w) = \mathcal{R}_{\text{eff}}^{G\times P_2}(u^{(A)}, w^{(A)}) \leq \mathcal{R}_{\text{eff}}^{G\times P_2}(u^{(A)}, v^{(B)}) + \mathcal{R}_{\text{eff}}^{G\times P_2}(v^{(B)}, w^{(A)}). \tag{10}$$

Using the preceding results, we design an algorithm with improved sampling count. It takes any procedure that produces effective resistance distribution that dominates the true one (call this an EREstimator), and produces samples that suffice for simulating the ideal sampling algorithm on G^k (cf. Subsection 2.3, Algorithm 1). The pseudocode for this routine is shown in Algorithm 2.

Note that from the perspective of this framework by picking edges with probabilities proportional to $\boldsymbol{w}_e \widetilde{\boldsymbol{r}}_e$, and extending them into walks, the previous result [7] can be viewed as utilizing a simple EREstimator that returns $\widetilde{\boldsymbol{r}}_e = 1/\boldsymbol{w}_e$ as the effective resistance of every edge.

▶ **Theorem 10.** *Given any graph $G = (V, E, \boldsymbol{w})$, any values of k and ε, and any effective resistance estimation algorithm* EREstimator *that produces w.h.p. estimates $\widetilde{\boldsymbol{r}}_e \geq \mathcal{R}_{\text{eff}}^{G}(e)$ for every edge $e \in E$, then calling* $\text{SparsifyG}^{\text{k}}(G, k, \varepsilon, \text{EREstimator})$ *produces an ε-sparsifier of G^k with $O(\varepsilon^{-2} k \langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle \log n)$ edges in time proportion to the cost of one call to* EREstimator *on a graph of twice the size, plus an overhead of $O(m + \varepsilon^{-2} k^2 \langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle \log^2 n)$.*

Proof. By Lemma 2, it suffices to show that this algorithm simulates the ideal sampling algorithm given in Algorithm 1. Once again we split into the cases of k being odd or even.

Algorithm 2 SparsifyG$^{\mathrm{k}}$ $(G, k, \varepsilon, \text{ERESTIMATOR})$

Input: Graph $G = (V, E, w)$, integer k, error ε, routine ERESTIMATOR that estimates upper bounds for effective resistances of a graph G.
Output: An ε-sparsifier of G^k

```
1. If $k$ is odd
   a. set $\widetilde{\boldsymbol{r}} \leftarrow \text{ERESTIMATOR}(G)$,
2. else $k$ is even
   a. Set $\widetilde{\boldsymbol{r}}^{(2)} \leftarrow \text{ERESTIMATOR}(G \times P_2)$,
   b. Set $\widetilde{\boldsymbol{r}}_e \leftarrow \widetilde{\boldsymbol{r}}^{(2)}(u^{(A)}, v^{(B)})$, for every edge $e = uv \in E[G]$        (cf. Lemma 9).
3. Set sampling overhead $h \leftarrow O(\varepsilon^{-2} \log n)$ and number of samples $N \leftarrow h \cdot k \cdot \langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle$.
4. Repeat $N$ times
   a. Pick an edge $e$ in $G$ with probability proportional to $\boldsymbol{w}_e \widetilde{\boldsymbol{r}}_e$.
   b. Pick an integer $0 \leq i < k$ uniformly at random, set $u_i$ and $u_{i+1}$ to be endpoints of $e$.
   c. Perform a random walk by taking $i$ steps from $u_i$ and $k - 1 - i$ steps from $u_{i+1}$.
   d. Add the edge $(u_0, u_1, \ldots, u_k)$ to $H$ with weight $1/(h \cdot \sum_{0 \leq i < k} \widetilde{\boldsymbol{r}}_{u_i u_{i+1}})$.
```

When k is odd, Lemma 7 implies that a walk $(u_0, u_1, \ldots, u_k)$ is sampled with probability proportional to $\boldsymbol{w}_{(u_0, u_1, \ldots, u_k)} \sum_{0 \leq i < k} \widetilde{\boldsymbol{r}}_{u_i, u_{i+1}}$, where $\widetilde{\boldsymbol{r}}_{u_i, u_{i+1}} \geq \mathcal{R}^G_{\text{eff}}(u_i, u_{i+1})$. Summing over all walks with fixed endpoints (u_0, u_k), by combining (2), (4) and Fact 6, this summation dominates the product $\boldsymbol{w}^{G^k}_{u_0, u_k} \mathcal{R}^{G^k}_{\text{eff}}(u_0, u_k)$. Thus, by Lemma 2 the resulting probability distribution satisfies the statement. The running time and the number of edges in the output sparsifier follow from Lemma 8.

In the case of k being even, by combining Lemma 5 and Lemma 9, we have

$$\mathcal{R}^{G^k}_{\text{eff}}(u, v) \leq \mathcal{R}^{G \times P_2}_{\text{eff}}\left(u^{(A)}, v^{(A)}\right) = \mathcal{R}^{G \times P_2}_{\text{eff}}\left(u^{(B)}, v^{(B)}\right).$$

Also, note that because k is even, each k step walk in G also corresponds to a walk in $G \times P_2$ that starts/ends on the same side, but alternates sides at each step. Using (10) and the symmetry between $u^{(A)} v^{(B)}$ and $u^{(B)} v^{(A)}$, it suffices to sample length k walks with estimated effective resistances satisfying $\widetilde{\boldsymbol{r}}_{uv} \geq \boldsymbol{r}^{G \times P_2}\left(u^{(A)}, v^{(B)}\right)$, for every edge $(u, v) \in G$. The rest of the algorithm follows similarly as in the case of odd k.

To enable picking a neighbor randomly, we need $O(\deg(v))$ preprocessing time for every vertex v, which implies a total preprocessing time of $O(m)$. The extra $O(k \log n)$ in the runtime overhead accounts for performing a random walk of length k, i.e. after preprocessing, a neighboring edge can be sampled using binary search in $O(\log n)$ time. ◀

This reduces the task of sampling edges in G^k to compute good upper bounds for the effective resistances of either the original graph G or of its double cover $G \times P_2$. In the next section we discuss this routine, with focus on density-independent routines.

4 Faster Density Independent Sparsification of Graphs

The monomial sparsification routine from the previous section only requires a distribution that dominates effective resistances for a given graph G. Additionally, we only need a good approximator of G to efficiently compute these approximate effective resistances. The major challenge in keeping the routine density independent is that most numerically oriented approaches for estimating effective resistances require $O(m \log n)$ time. Instead, a more relevant approach is to utilize "low stretch spanning trees".

Given a graph $G = (V, E, \boldsymbol{w})$, and a tree T, we define the stretch of an edge $e = (u, v) \in E$ w.r.t. T as the ratio of the total resistance on the unique path $\mathcal{P}_T(e)$ between u and v in T to the resistance of e:

$$str_{T,G}(e) \stackrel{\text{def}}{=} \boldsymbol{w}_e \sum_{e' \in \mathcal{P}_T(e)} \frac{1}{\boldsymbol{w}_{e'}}.$$

Extending this definition, the stretch of a subgraph $G'(V', E')$ of G w.r.t. T is given by

$$str_{T,G}(G') \stackrel{\text{def}}{=} \sum_{e \in E'} str_{T,G}(e).$$

We will drop the usage of the second term in the subscript when the underlying graph is obvious from the context.

The advantage of using trees with respect to whom G has low stretch is that the resistance of the path between vertices u and v in the tree can be used as an estimate for the effective resistance of (u, v), and more importantly, the stretch of all edges can be computed using lowest common ancestor queries in only $O(m)$ time [19]. In this context, Lemma 2 can be rewritten as:

▶ **Lemma 11.** *If we have a tree $T \preceq G$, then we can construct an ε-sparsifier of G with $O(\varepsilon^{-2} str_T(G) \log n)$ edges in $O(m)$ time.*

However, we are still left with the issue of constructing such a tree. Abraham and Neiman [2] showed that a tree with stretch $\widehat{O}(m \log n)$ can be constructed in time $\widehat{O}(m \log n)$. This running time does not help our goal of being density-independent. Also, the average stretch is not low enough for the stretches to serve as effective resistance estimates. To tackle both of these issues, we follow the approach used in [26]. We present now a brief overview of this approach and we include the details in Appendix B.

1. Construct a tree T and a graph $\widehat{G}$ obtained by removing $O(m/\log n)$ edges from G such that $str_T(\widehat{G}) \leq \widehat{O}(m \log n)$. This can be computed in $\widehat{O}(m)$ time, using [10, Lemma 5.9] applied with $k = O(\log n)$.
2. Sparsify the removed edges in $O(m)$ time using any standard sparsification method [27, 28] to get H'.
3. To sparsify $\widehat{G}$, construct a series of graphs $\widehat{G}^{(0)}, \widehat{G}^{(1)}, \ldots, \widehat{G}^{(\tau)}$, where $\widehat{G}^{(0)} = \widehat{G}$ and $\widehat{G}^{(\tau)}$ is a graph with low enough stretch such that an $O(1)$-sparsifier $\widehat{H}^{(\tau)}$ of $\widehat{G}^{(\tau)}$ can be constructed in $O(m)$ time.
4. Use the sparsifier $\widehat{H}^{(\tau)}$ to construct an $O(1)$-sparsifier $\widehat{H}^{(\tau-1)}$ of $\widehat{G}^{(\tau-1)}$ and so on, until we get an $O(1)$-sparsifier $\widehat{H}^{(1)}$ of $\widehat{G}^{(1)}$. Every sparsifier $\widehat{H}^{(i)}$ has at most $O(n \log n)$ edges.
5. Repeating Step 4 a final time using effective resistance upper bounds computed from $\widehat{H}^{(0)}$, we compute an ϵ-sparsifier $\widehat{H}$ of $\widehat{G}$. Bringing in the small ϵ only at the last step, allows us to keep the accuracy-related overhead in the intermediate steps at $O(1)$.

This gives us the following results:

▶ **Lemma 12.** *There is a routine that takes a weighted undirected graph G with n vertices, m edges, an error $\epsilon > 0$, and produces in $\widehat{O}(m + \varepsilon^{-2} n \log^4 n)$ time an ϵ-sparsifier of G with $O(\epsilon^{-2} n \log n)$ edges, as well as effective resistance upper bounds $\widetilde{\boldsymbol{r}}$ such that $\langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle = \widehat{O}(n \log^2 n)$.*

▶ **Corollary 13.** *There is a combinatorial algorithm that for any graph G on n vertices and m edges, and any error $\epsilon > 0$, produces in $\widehat{O}(m + n\log^6 n)$ time an ϵ-sparsifier of G with $\widehat{O}(\varepsilon^{-2} n \log^2 n)$ edges, as well as effective resistance upper bounds $\widetilde{\boldsymbol{r}}$ such that $\langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle = \widehat{O}(n \log^3 n)$.*

The current fastest sparsification routines compute effective resistances via the Johnson-Lindenstrauss transform [35], which in turn requires the use of fast linear system solvers [27].

▶ **Lemma 14.** *Given a graph G, we can compute 2-approximations to its effective resistances in $\widehat{O}(m \log n + n \log^2 n)$ time.*

This runtime bound can be obtained by letting the depth approach n in the proof of Theorem 1.2 on page 49 of [27]. The effective resistances can in turn be extracted from the call to Sparsify made at $i = 0$ in the pseudocode in Figure 11 on page 46. We omit details on these steps in the hope that significantly simpler sparsification routines with similar performances will be developed.

Now, we can prove our main result.

Proof of Theorem 1. The upper bound on effective resistances obtained by Lemma 12, when combined with Theorem 10 produces an ϵ-sparsifier of G^k with $\widehat{O}(\epsilon^{-2} kn \log^3 n)$ edges in $\widehat{O}(m + \varepsilon^{-2} k^2 n \log^4 n)$ time. Sparsifying this graph once again using Lemma 14 then leads to the main result as stated in Theorem 1. ◀

The combinatorial guarantees follow similarly from Corollary 13.

Acknowledgements. We thank David Durfee for the various discussions related to applications of these ideas in [14], and the anonymous reviewers of previous versions of this manuscript for their very helpful comments and suggestions.

References

1 Ittai Abraham, David Durfee, Ioannis Koutis, Sebastian Krinninger, and Richard Peng. On fully dynamic graph sparsifiers. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 335–344. IEEE, 2016. Available at: `http://arxiv.org/abs/1604.02094`.

2 Ittai Abraham and Ofer Neiman. Using petal-decompositions to build a low stretch spanning tree. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 395–406. ACM, 2012. Available at: `https://www.microsoft.com/en-us/research/wp-content/uploads/2012/01/spanning-full1.pdf`.

3 Sayan Bhattacharya, Monika Henzinger, and Giuseppe F Italiano. Deterministic fully dynamic data structures for vertex cover and matching. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 785–804. SIAM, 2014. Available at: `https://arxiv.org/abs/1412.1318`.

4 Andreas Björklund, Rasmus Pagh, Virginia Vassilevska Williams, and Uri Zwick. Listing triangles. In *International Colloquium on Automata, Languages, and Programming*, pages 223–234. Springer, 2014.

5 A. Borodin and I. Munro. *The computational complexity of algebraic and numeric problems.* American Elsevier Pub. Co New York, 1975.

6 Karl Bringmann and Konstantinos Panagiotou. Efficient sampling methods for discrete distributions. In Artur Czumaj, Kurt Mehlhorn, Andrew Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming: 39th International Colloquium, ICALP*

2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I, pages 133–144. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. doi:10.1007/978-3-642-31594-7_12.

7 Dehua Cheng, Yu Cheng, Yan Liu, Richard Peng, and Shang-Hua Teng. Efficient sampling for Gaussian graphical models via spectral sparsification. *Proceedings of The 28th Conference on Learning Theory*, pages 364–390, 2015. Available at http://jmlr.org/proceedings/papers/v40/Cheng15.pdf.

8 Yu Cheng and Dehua Cheng. Personal Communication, 2016.

9 Michael B. Cohen, Yin Tat Lee, Cameron Musco, Christopher Musco, Richard Peng, and Aaron Sidford. Uniform sampling for matrix approximation. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, ITCS'15, pages 181–190, New York, NY, USA, 2015. ACM. doi:10.1145/2688073.2688113.

10 Michael B. Cohen, Gary L. Miller, Jakub W. Pachocki, Richard Peng, and Shen Chen Xu. Stretching stretch. *arXiv preprint arXiv:1401.2454*, 2014. Available at: https://arxiv.org/abs/1401.2454.

11 Michael B. Cohen, Cameron Musco, and Christopher Musco. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1758–1777, 2017. doi:10.1137/1.9781611974782.115.

12 Peter G. Doyle and J. Laurie Snell. *Random Walks and Electric Networks*, volume 22 of *Carus Mathematical Monographs*. Mathematical Association of America, 1984. Available at: https://arxiv.org/abs/math/0001057.

13 David Durfee, Rasmus Kyng, John Peebles, Anup B. Rao, and Sushant Sachdeva. Sampling random spanning trees faster than matrix multiplication. *CoRR*, abs/1611.07451, 2016. Available at: http://arxiv.org/abs/1611.07451.

14 David Durfee, John Peebles, Richard Peng, and Anup B. Rao. Determinant-preserving sparsification of SDDM matrices with applications to counting and sampling spanning trees. *CoRR*, abs/1705.00985, 2017. URL: http://arxiv.org/abs/1705.00985.

15 Michael L. Fredman and Robert Endre Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *J. ACM*, 34:596–615, July 1987.

16 Andrew V. Goldberg and Robert E. Tarjan. Efficient maximum flow algorithms. *Communications of the ACM*, 57(8):82–89, 2014. Available at: http://cacm.acm.org/magazines/2014/8/177011-efficient-maximum-flow-algorithms/fulltext.

17 Gramoz Goranci, Monika Henzinger, and Mikkel Thorup. Incremental exact min-cut in poly-logarithmic amortized update time. In Piotr Sankowski and Christos D. Zaroliagis, editors, *24th Annual European Symposium on Algorithms, ESA 2016, August 22-24, 2016, Aarhus, Denmark*, volume 57 of *LIPIcs*, pages 46:1–46:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. Full version available at: https://arxiv.org/abs/1611.06500. doi:10.4230/LIPIcs.ESA.2016.46.

18 Manoj Gupta and Richard Peng. Fully dynamic (1+ e)-approximate matchings. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 548–557, 2013. doi:10.1109/FOCS.2013.65.

19 Dov Harel and Robert Endre Tarjan. Fast algorithms for finding nearest common ancestors. *siam Journal on Computing*, 13(2):338–355, 1984.

20 Nick Harvey. Matrix concentration and sparsification. Workshop on "Randomized Numerical Linear Algebra (RandNLA): Theory and Practice", 2012. Available at: http://www.drineas.org/RandNLA/slides/Harvey_RandNLA@FOCS_2012.pdf.

21 Monika Henzinger, Sebastian Krinninger, and Danupon Nanongkai. Sublinear-time decremental algorithms for single-source reachability and shortest paths on directed graphs. In

Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing, STOC'14, pages 674–683, 2014. Available at: https://arxiv.org/abs/1504.07959.

22 Gorav Jindal and Pavel Kolev. Faster spectral sparsification of laplacian and SDDM matrix polynomials. *CoRR*, abs/1507.07497, 2015. Available at: http://arxiv.org/abs/1507.07497.

23 Michael Kapralov and Rina Panigrahy. Spectral sparsification via random spanners. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 393–398. ACM, 2012. Available at: https://www.microsoft.com/en-us/research/wp-content/uploads/2012/01/sig-alternate.pdf.

24 Donald E. Knuth. *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms.* Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1997.

25 Ioannis Koutis. Simple parallel and distributed algorithms for spectral graph sparsification. In *Proceedings of the 26th ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA'14, pages 61–66, New York, NY, USA, 2014. ACM. Available at: http://arxiv.org/abs/1402.3851. doi:10.1145/2612669.2612676.

26 Ioannis Koutis, Alex Levin, and Richard Peng. Faster spectral sparsification and numerical algorithms for SDD matrices. *ACM Trans. Algorithms*, 12(2):17:1–17:16, December 2015.

27 Rasmus Kyng, Yin Tat Lee, Richard Peng, Sushant Sachdeva, and Daniel A. Spielman. Sparsified cholesky and multigrid solvers for connection laplacians. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 842–850. ACM, 2016. Available at http://arxiv.org/abs/1512.01892.

28 Rasmus Kyng, Jakub Pachocki, Richard Peng, and Sushant Sachdeva. A framework for analyzing resparsification algorithms. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'17, pages 2032–2043, Philadelphia, PA, USA, 2017. Society for Industrial and Applied Mathematics. Available at: https://arxiv.org/abs/1611.06940.

29 Rasmus Kyng and Sushant Sachdeva. Approximate gaussian elimination for laplacians-fast, sparse, and simple. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 573–582. IEEE, 2016. Available at: https://arxiv.org/abs/1605.02353.

30 László Lovász. Random walks on graphs: A survey, 1993. Available at: http://www.cs.elte.hu/~lovasz/erdos.pdf.

31 Rasmus Pagh and Charalampos E. Tsourakakis. Colorful triangle counting and a mapreduce implementation. *Information Processing Letters*, 112(7):277–281, 2012.

32 Richard Peng and Daniel A. Spielman. An efficient parallel solver for SDD linear systems. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC'14, pages 333–342, New York, NY, USA, 2014. ACM. Available at http://arxiv.org/abs/1311.3286.

33 Franco P. Preparata and Michael I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag New York, Inc., New York, NY, USA, 1985.

34 Christian Sommer. Shortest-path queries in static networks. *ACM Computing Surveys (CSUR)*, 46(4):45, 2014. Available at: http://www.shortestpaths.com/spq-survey.pdf.

35 D. Spielman and N. Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011. doi:10.1137/080734029.

36 Daniel A. Spielman. Lecture notes on graphs and networks, October 2007. Available at: http://www.cs.yale.edu/homes/spielman/462/2007/lect9-07.pdf.

37 Joel A. Tropp. User-friendly tail bounds for sums of random matrices. *Found. Comput. Math.*, 12(4):389–434, August 2012. doi:10.1007/s10208-011-9099-z.

38 Charalampos E. Tsourakakis. Fast counting of triangles in large real networks without counting: Algorithms and laws. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 608–617. IEEE, 2008. Available at `http://people.seas.harvard.edu/~babis/tsourICDM08.pdf`.

39 A. J. Walker. New fast method for generating discrete random numbers with arbitrary frequency distributions. *Electronics Letters*, 10(8):127–128, April 1974. `doi:10.1049/el:19740097`.

40 David P. Woodruff et al. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014. Available at: `http://researcher.watson.ibm.com/researcher/files/us-dpwoodru/journal.pdf`.

A Omitted Proofs For Section 3

We give here some additional details on lemmas from Section 3 that are direct consequences of steps in previous works. The total summation of the sampling weights follows from a summation identical to the special case of uniform sampling, as presented in [7, Lemma 29]. More precisely, it is done by evaluating the total weights of all walks with a fixed edge $e \in G$.

Proof of Lemma 8. We first show by induction that the total weights of all length k walks whose i^{th} edge is e is exactly $\boldsymbol{w}_e$.

The base case of $k = 1$ is trivial as only e is a length 1 walk between u_0 and u_1.

The inductive case of $k > 1$ has two cases: $i > 0$ or $i < k - 1$. We consider only the case $i > 0$, as the other one follows by symmetry. Expanding the weight of a length k walk gives:

$$\boldsymbol{w}_{(u_0,u_1,\ldots,u_k)} = \boldsymbol{w}_{(u_0,u_1,\ldots,u_{k-1})} \frac{\boldsymbol{A}_{u_{k-1}u_k}}{\boldsymbol{d}_{u_{k-1}}}.$$

The fact that $i < k - 1$ means that u_k can be any neighbor of u_{k-1}, leading to a sum that cancels the $\boldsymbol{d}_{u_{k-1}}$ term in the denominator. The result then follows from the inductive hypothesis applied to walks of length $k - 1$ that have edge e indexed as the i^{th} walk step:

$$\begin{aligned}\sum_{\substack{(u_0,u_1,\ldots,u_k)\\ e=(u_i,u_{i+1})}} \boldsymbol{w}_{(u_0,u_1,\ldots,u_k)} &= \sum_{\substack{(u_0,u_1,\ldots,u_k)\\ e=(u_i,u_{i+1})}} \boldsymbol{w}_{(u_0,u_1,\ldots,u_{k-1})} \sum_{u_k} \frac{\boldsymbol{A}_{u_{k-1}u_k}}{\boldsymbol{d}_{u_{k-1}}} \\ &= \sum_{\substack{(u_0,u_1,\ldots,u_{k-1})\\ e=(u_i,u_{i+1})}} \boldsymbol{w}_{(u_0,u_1,\ldots,u_{k-1})} \overset{\text{By I.H.}}{=} \boldsymbol{w}_e.\end{aligned}$$

The proof then uses a double counting argument that breaks the summation over edges $e = (u_i, u_{i+1}) \in G$, so as the original summation in (8) becomes equivalent to

$$\sum_{e\in G} \widetilde{\boldsymbol{r}}_e \sum_{0\le i<k} \sum_{\substack{(u_0,u_1,\ldots,u_k)\\ e=(u_i,u_{i+1})}} \boldsymbol{w}_{(u_0,u_1,\ldots,u_k)} = \sum_{e\in G} \widetilde{\boldsymbol{r}}_e \cdot k\boldsymbol{w}_e = k \left\langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \right\rangle.$$ ◀

Before we establish an equivalence relation between the effective resistances of the graphs G^2 and $G \times P_2$, we need some notation.

▶ **Definition 15** (Schur Complement). Let $\boldsymbol{M} = \begin{pmatrix} \boldsymbol{M}_{[F,F]} & \boldsymbol{M}_{[F,C]} \\ \boldsymbol{M}_{[C,F]} & \boldsymbol{M}_{[C,C]} \end{pmatrix}$ be a symmetric matrix. The Schur Complement of $\boldsymbol{M}$ induced by removing the block F is defined by

$$Sc(\boldsymbol{M}, F) \overset{\text{def}}{=} \boldsymbol{M}_{[C,C]} - \boldsymbol{M}_{[C,F]} \boldsymbol{M}_{[F,F]}^{-1} \boldsymbol{M}_{[F,C]}.$$

It is known that for any Laplacian $\boldsymbol{M}$ of a graph G, $Sc(M, F)$ is the Laplacian of a graph G^C which is formed by the following iterative process:

- For every vertex $u \in F$
  - For every pair of edges uv_1 and uv_2 in the current graph (with edges from prior steps)
    - Delete edges uv_1 and uv_2, and add a new edge v_1v_2 with weight $\boldsymbol{w}_{uv_1}\boldsymbol{w}_{uv_2}/\boldsymbol{d}_u$, where $\boldsymbol{d}_u$ is the weighted degree of u w.r.t. the current graph.
  - Delete vertex u.

▶ **Lemma 16.** *For every vector* $\boldsymbol{z} = \begin{pmatrix} \boldsymbol{z}_1 \\ 0 \end{pmatrix}$ *it holds that*

$$\boldsymbol{z}_1^T \left(\boldsymbol{D} - \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{A}\right)^\dagger z_1 = \begin{pmatrix} \boldsymbol{z}_1^T & 0^T \end{pmatrix} \begin{pmatrix} \boldsymbol{D} & -\boldsymbol{A} \\ -\boldsymbol{A} & \boldsymbol{D} \end{pmatrix}^\dagger \begin{pmatrix} \boldsymbol{z}_1 \\ 0 \end{pmatrix}.$$

By symmetry for any vector $\boldsymbol{z} = \begin{pmatrix} 0 \\ \boldsymbol{z}_2 \end{pmatrix}$ *it holds that*

$$\boldsymbol{z}_2^T \left(\boldsymbol{D} - \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{A}\right)^\dagger \boldsymbol{z}_2 = \begin{pmatrix} 0^T & \boldsymbol{z}_2^T \end{pmatrix} \begin{pmatrix} \boldsymbol{D} & -\boldsymbol{A} \\ -\boldsymbol{A} & \boldsymbol{D} \end{pmatrix}^\dagger \begin{pmatrix} 0 \\ \boldsymbol{z}_2 \end{pmatrix}.$$

In particular, the effective resistances are maintained under Schur complement.

Proof. Consider the linear system

$$\begin{pmatrix} \boldsymbol{D} & -\boldsymbol{A} \\ -\boldsymbol{A} & \boldsymbol{D} \end{pmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{pmatrix} = \begin{pmatrix} \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \end{pmatrix} \iff \begin{matrix} \boldsymbol{D}\boldsymbol{x} - \boldsymbol{A}y = \boldsymbol{z}_1 \\ -\boldsymbol{A}\boldsymbol{x} + \boldsymbol{D}y = \boldsymbol{z}_2` \end{matrix} \iff \begin{matrix} \boldsymbol{x} = \boldsymbol{D}^{-1}\left(\boldsymbol{z}_1 + \boldsymbol{A}\boldsymbol{y}\right) \\ \boldsymbol{y} = \boldsymbol{D}^{-1}\left(\boldsymbol{z}_2 + \boldsymbol{A}\boldsymbol{x}\right) \end{matrix}.$$

Since $\boldsymbol{z}_2 = 0$, we have

$$\begin{matrix} \boldsymbol{D}\boldsymbol{x} = \boldsymbol{z}_1 + \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{x} \\ \boldsymbol{y} = \boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{x} \end{matrix} \implies \boldsymbol{x} = \left(\boldsymbol{D} - \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{A}\right)^\dagger \boldsymbol{z}_1.$$

and thus

$$\begin{pmatrix} \boldsymbol{z}_1^T & \boldsymbol{z}_2^T \end{pmatrix} \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{pmatrix} = \boldsymbol{z}_1^T\boldsymbol{x} = \boldsymbol{z}_1^T \left(\boldsymbol{D} - \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{A}\right)^\dagger \boldsymbol{z}_1.$$ ◀

We can now prove that the effective resistance between u and v in G^2 is the same as the effective resistance between $u^{(A)}$ and $v^{(A)}$ in $G \times P_2$

Proof of Lemma 9. Notice that

$$\boldsymbol{L}_{G^2} = \boldsymbol{D} - \boldsymbol{A}\boldsymbol{D}^{-1}\boldsymbol{A}$$

is the Schur Complement of

$$\boldsymbol{L}_{G\times P_2} = \begin{pmatrix} \boldsymbol{D} & -\boldsymbol{A} \\ -\boldsymbol{A} & \boldsymbol{D} \end{pmatrix}$$

with respect to one half of the vertices, e.g. $V^{(B)}$. The statement follows by Lemma 16. ◀

B Omitted Proofs For Section 4

The following is a detailed exposition of the techniques used to achieve density independent sparsification of a given graph G. The ideas are mainly from [26], but the arguments are tailored to our setting. For the reader's convenience, we present again the scheme overview:

1. Construct a tree T and a graph $\widehat{G}$ obtained by removing $O(m/\log n)$ edges from G such that $str_T(\widehat{G}) \leq \widehat{O}(m \log n)$. This can be computed in $\widehat{O}(m)$ time, using [10, Lemma 5.9] applied with $k = O(\log n)$.
2. Sparsify the removed edges in $O(m)$ time using any standard sparsification method [27, 28] to get H'.
3. To sparsify $\widehat{G}$, construct a series of graphs $\widehat{G}^{(0)}, \widehat{G}^{(1)}, \ldots, \widehat{G}^{(\tau)}$, where $\widehat{G}^{(0)} = \widehat{G}$ and $\widehat{G}^{(\tau)}$ is a graph with low enough stretch such that an $O(1)$-sparsifier $\widehat{H}^{(\tau)}$ of $\widehat{G}^{(\tau)}$ can be constructed in $O(m)$ time.
4. Use the sparsifier $\widehat{H}^{(\tau)}$ to construct an $O(1)$-sparsifier $\widehat{H}^{(\tau-1)}$ of $\widehat{G}^{(\tau-1)}$ and so on, until we get an $O(1)$-sparsifier $\widehat{H}^{(1)}$ of $\widehat{G}^{(1)}$. Every sparsifier $\widehat{H}^{(i)}$ has at most $O(n \log n)$ edges.
5. Repeating Step 4 a final time using effective resistance upper bounds computed from $\widehat{H}^{(0)}$, we compute an ϵ-sparsifier $\widehat{H}$ of $\widehat{G}$. Bringing in the small ϵ only at the last step, allows us to keep the accuracy-related overhead in the intermediate steps at $O(1)$.

B.1 Proof Of Lemma 12

We give now a detailed description of Step 3. The i^{th} graph $\widehat{G}^{(i)}$ in the series is defined by

$$\widehat{G}^{(i)} = \widehat{G} + 2^i \cdot T.$$

We establish next an upper bound on the graph stretch $str_T(\widehat{G}^{(i)})$, for every i. Our proof uses the following notation that highlights the relation between edge stretch and edge weight function.

By definition, the stretch of any tree edge equals 1 and the "on-tree" stretch $str_{T,\widehat{G}^{(i)}}(T)$ has value $n-1$. On the other hand, the stretch of every non-tree edge $e \in \widehat{G}^{(i)} \backslash T$ satisfies

$$str_{T,\widehat{G}^{(i)}}(e) = \boldsymbol{w}_e^{\widehat{G}^{(i)}} \sum_{e' \in \mathcal{P}_T(e)} \left(\boldsymbol{w}_{e'}^{\widehat{G}^{(i)}}\right)^{-1} = \boldsymbol{w}_e^{\widehat{G}} \sum_{e' \in \mathcal{P}_T(e)} \left((2^i+1)\boldsymbol{w}_{e'}^{\widehat{G}}\right)^{-1} \leq 2^{-i} \cdot str_{T,\widehat{G}}(e).$$

Moreover, since

$$str_{T,\widehat{G}^{(i)}}(\widehat{G}\backslash T) \leq 2^{-i} \cdot str_{T,\widehat{G}}(\widehat{G}\backslash T) \leq 2^{-i} \cdot str_{T,\widehat{G}}(\widehat{G}) = O(2^{-i} \cdot m \log n),$$

it follows that the total stretch of graph $\widehat{G}^{(i)}$ w.r.t. T is bounded by

$$str_T(\widehat{G}^{(i)}) = str_{T,\widehat{G}^{(i)}}(\widehat{G}\backslash T) + str_{T,\widehat{G}^{(i)}}(T) \leq O(2^{-i} \cdot m \log n).$$

Therefore, the initial graph $\widehat{G}^{(\tau)}$ for $\tau = \Omega(\log\log n)$ has total stretch

$$str_T(\widehat{G}^{(\tau)}) = \widehat{O}(m/\log^2 n).$$

Using Lemma 11, we can compute in $O(m)$ time an $O(1)$-sparsifier $G'^{(\tau)}$ of $\widehat{G}^{(\tau)}$ with $\widehat{O}(m/\log n)$ edges. Invoking any standard nearly-linear time sparsification algorithm on $G'^{(\tau)}$ then gives us in $O(m)$ time a $O(1)$-sparsifier $\widehat{H}^{(\tau)}$ of $G^{(\tau)}$ with $O(n \log n)$ edges.

We present now the TreeSparsify routine which is used in Step 4 and Step 5.

```
Algorithm 3 TreeSparsify($G, G', \kappa, \varepsilon$)
Input: Graph $G = (V, E, \boldsymbol{w})$ with $\kappa$-sparsifier $G'$, and error $\varepsilon > 0$.
Output: $\widetilde{G}$ that is an $\varepsilon$-sparsifier of $G$.

1. Compute a low stretch spanning tree $T$ of $G'$.
2. Compute an upper bound on all leverage scores $\widetilde{\boldsymbol{\tau}}$ of $G$ using [19].
3. Sample $O(\varepsilon^{-2} str_T(G) \log n)$ edges of $G$ using IdealSample($G, \varepsilon, \widetilde{\boldsymbol{\tau}}$) (cf. Algorithm 1).
```

▶ **Lemma 17.** *Given a κ-sparsifier G' of G and $\varepsilon > 0$,* TreeSparsify$(G, G', \kappa, \varepsilon)$ *produces an ε-sparsifier of G with $\widehat{O}(\varepsilon^{-2}\kappa\, |E(G')| \log^2 n)$ edges in $\widehat{O}(m + \varepsilon^{-2}\kappa\, |E(G')| \log^3 n)$ time.*

Proof. To apply Lemma 2, we have to compute a vector $\widetilde{\boldsymbol{r}} \geq \mathcal{R}_{\text{eff}}^{G}$ and give an upper bound on $\langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle$. Since $\boldsymbol{L}_G \preceq \boldsymbol{L}_{G'}$, by [26, Lemma 6.4] we have $str_T(G) \leq str_T(G')$. Additionally, since $\boldsymbol{L}_T \preceq \boldsymbol{L}_{G'} \preceq \kappa \boldsymbol{L}_G$, it follows that

$$\widetilde{\boldsymbol{r}} \stackrel{\text{def}}{=} \kappa \cdot \mathcal{R}_{\text{eff}}^{T} \geq \mathcal{R}_{\text{eff}}^{G}. \tag{11}$$

Using the above statements, and the low stretch spanning tree construction of Abraham and Neiman [2], we obtain

$$\langle \boldsymbol{w}, \widetilde{\boldsymbol{r}} \rangle = \kappa \cdot str_T(G) \leq \kappa \cdot str_T(G') = \widehat{O}(\kappa\, |E(G')| \log n).$$

The statement follows by Lemma 2. ◀

We present now the core iterative procedure underlying Step 4 and Step 5:

(i) Let $\delta > 0$ be an error parameter. In Step 4, we set $\delta = O(1)$, whereas $\delta = \epsilon$ in Step 5.

(ii) Straightforward checking shows that by construction $\widehat{H}^{(i+1)}$ is a $O(1)$-sparsifier of $\widehat{H}^{(i)}$.

(iii) Compute a $\delta/2$-sparsifier $G'^{(i)}$ of $\widehat{G}^{(i)}$ with $\widehat{O}(\delta^{-2} n \log^3 n)$ edges in $\widehat{O}(m + \delta^{-2} n \log^4 n)$ time, calling TreeSparsify$(\widehat{G}^{(i)}, \widehat{H}^{(i+1)}, O(1), \delta)$. The guarantees follow by Lemma 17.

(iv) Compute a $\delta/2$-sparsifier $\widehat{H}^{(i)}$ of $G'^{(i)}$ with $O(\delta^{-2} n \log n)$ edges in $\widehat{O}(\delta^{-2} n \log^4 n)$ time, using Lemma 14 and Lemma 2. Thus, $\widehat{H}^{(i)}$ is a δ-sparsifier of $\widehat{G}^{(i)}$.

We analyze now the runtime of Steps 4 and 5. In Step 4, there are $O(\log\log n)$ calls to TreeSparsify each with $\delta = O(1)$. Thus, by Lemma 17, Step 4 runs in $\widehat{O}(m + n \log^4 n)$ time. In Step 5, we set $\delta = \epsilon$. Then, by Lemma 14 and Lemma 2, the runtime of Step 5 is bounded by $\widehat{O}(m + \epsilon^{-2} n \log^4 n)$.

B.2 Proof Of Corollary 13

We use purely combinatorial constructions of graph sparsifiers that are based on spanners [23, 25, 28]. We summarize these results in the following lemma.

▶ **Lemma 18** ([28, Theorem 4.1])**.** *Given G and error $\varepsilon > 0$, we can compute an ε-spectral sparsifier of G with $\widehat{O}(n \log^2 n)$ edges in $\widehat{O}(m \log^2 n)$ time.*

We show now that the algorithm in Lemma 18 applied to our sparsification scheme yields Corollary 13. We argue in a similar manner as in the routine calling numerical sparsifiers, outlined in Corollary 12. Here, in contrast, every sparsifier $\widehat{H}^{(i+1)}$ has $\widehat{O}(n \log^2 n)$ edges, and thus every sparsifier $G'^{(i)}$ has $\widehat{O}(n \log^4 n)$ edges. Hence, every consecutive re-sparsification call yield a sparsifier $\widehat{H}^{(i)}$ with $\widehat{O}(n \log^2 n)$ edges in $\widehat{O}(n \log^6 n)$ time.

Maximum Matching in Two, Three, and a Few More Passes over Graph Stream

Sagar Kale[1] and Sumedh Tirodkar[2]

1 Department of Computer Science, Dartmouth College, Hanover, NH, USA
sag@cs.dartmouth.edu

2 School of Technology and Computer Science, TIFR, Mumbai, India
sumedh.tirodkar@tifr.res.in

Abstract

We consider the maximum matching problem in the semi-streaming model formalized by Feigenbaum, Kannan, McGregor, Suri, and Zhang [13] that is inspired by giant graphs of today. As our main result, we give a two-pass $(1/2 + 1/16)$-approximation algorithm for triangle-free graphs and a two-pass $(1/2 + 1/32)$-approximation algorithm for general graphs; these improve the approximation ratios of $1/2 + 1/52$ for bipartite graphs and $1/2 + 1/140$ for general graphs by Konrad, Magniez, and Mathieu [20]. In three passes, we are able to achieve approximation ratios of $1/2 + 1/10$ for triangle-free graphs and $1/2 + 1/19.753$ for general graphs. We also give a multi-pass algorithm where we bound the number of passes *precisely* – we give a $(2/3 - \varepsilon)$-approximation algorithm that uses $2/(3\varepsilon)$ passes for triangle-free graphs and $4/(3\varepsilon)$ passes for general graphs. Our algorithms are simple and combinatorial, use $O(n \log n)$ space, and (can be implemented to) have $O(1)$ update time per edge.

For general graphs, our multi-pass algorithm improves the best known *deterministic* algorithms in terms of the number of passes:

- Ahn and Guha [1] give a $(2/3 - \varepsilon)$-approximation algorithm that uses $O(\log(1/\varepsilon)/\varepsilon^2)$ passes, whereas our $(2/3 - \varepsilon)$-approximation algorithm uses $4/(3\varepsilon)$ passes;
- they also give a $(1-\varepsilon)$-approximation algorithm that uses $O(\log n \cdot \mathrm{poly}(1/\varepsilon))$ passes, where n is the number of vertices of the input graph; although our algorithm is $(2/3-\varepsilon)$-approximation, our number of passes do not depend on n.

Earlier multi-pass algorithms either have a large constant inside big-O notation for the number of passes [9] or the constant cannot be determined due to the involved analysis [22, 1], so our multi-pass algorithm should use much fewer passes for approximation ratios bounded slightly below $2/3$.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Semi Streaming, Maximum Matching

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.15

1 Introduction

Maximum matching is a well-studied problem in a variety of computational models. We consider it in the semi-streaming model formalized by Feigenbaum, Kannan, McGregor, Suri, and Zhang [13] that is inspired by generation of ginormous graphs in recent times. A graph stream is an (adversarial) sequence of the edges of a graph, and a semi-streaming algorithm must access the edges in the given order and use $O(n \operatorname{polylog} n)$ space only, where n is the number of vertices; note that a matching can have size $\Omega(n)$, so $\Omega(n \log n)$ space is necessary. The number of times an algorithm goes over a stream of edges is called the number of *passes*. A trivial $(1/2)$-approximation algorithm that can be easily implemented as a one-pass

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 15; pp. 15:1–15:21
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

semi-streaming algorithm is to output a maximal matching. Since the formalization of the semi-streaming model more than a decade ago, the problem of finding a better than (1/2)-approximation algorithm or proving that one cannot do better has baffled researchers [21]. In a step towards resolving this, Goel, Kapralov, and Khanna [14] proved that for any $\varepsilon > 0$, a one-pass semi-streaming $(2/3 + \varepsilon)$-approximation algorithm does not exist; Kapralov [16], building on those techniques, showed non-existence of one-pass semi-streaming $(1 - 1/e + \varepsilon)$-approximation algorithms for any $\varepsilon > 0$. A natural next question is: Can we do better in, say, two passes or three passes? In answering that, Konrad, Magniez, and Mathieu [20] gave three-pass and two-pass algorithms that output matchings that are better than (1/2)-approximate. In this work, we give algorithms that improve their approximation ratios for two-pass and three-pass algorithms. We also give a multi-pass algorithm that does better than the best known multi-pass algorithms for at least initial few passes. We are able to bound the number of passes precisely: we give a $(2/3 - \varepsilon)$-approximation algorithm that uses $2/(3\varepsilon)$ passes for triangle-free graphs and $4/(3\varepsilon)$ passes for general graphs. Earlier works either have a large constant inside the big-O notation for the number of passes [9] or the constant cannot be determined due to the involved analysis [22, 1]. For example, the $(1 - \varepsilon)$-approximation algorithm by Eggert et al. [9] potentially uses $288/\varepsilon^5$ passes, and for the $(1-\varepsilon)$-approximation algorithms by McGregor [22] and Ahn and Guha [1], the constants inside the big-O bound cannot be determined due to the involved analysis. The $(2/3 - \varepsilon)$-approximation algorithm by Feigenbaum et al. [13] uses $O(\log(1/\varepsilon)/\varepsilon)$ passes, which is $O(\log(1/\varepsilon))$ factor larger than the number of passes we use to get the same approximation ratio. Our algorithms are simple and combinatorial, use $O(n \log n)$ space, and (can be implemented to) have $O(1)$ update time per edge. We also give an explicit and tight analysis of the three-pass algorithm by Konrad et al. [20] that is reminiscent of Feigenbaum et al.'s [13] multi-pass algorithm.

Technical overview

If we can find a matching M such that there are no augmenting paths of length 3 in $M \cup M^*$, where M^* is a maximum matching, then M is (2/3)-approximate, i.e., $(1/2 + 1/6)$-approximate. This is because, in each connected component of $M \cup M^*$, the ratio of M-edges to M^*-edges is at least 2/3. This is the basis for the $(2/3 - \varepsilon)$-approximation algorithm by Feigenbaum et al. [13] that uses $O(\log(1/\varepsilon)/\varepsilon)$ passes. The same idea is used by Konrad et al. [20] in the analysis of their two-pass algorithms. In the first pass, they find a maximal matching M_0 and some subset of support edges, say S. If M_0 is so bad that $M_0 \cup M^*$ is almost entirely made up of augmenting paths of length 3 (i.e., $|M_0| \approx |M^*|/2$), then by the end of the second pass, they manage to augment (using length-3 augmentations) a constant fraction of M_0 using S and a fresh access to the edges, resulting in a better than (1/2)-approximation. On the other hand, if M_0 is not so bad, then they already have a good matching. One limitation this idea faces is that a fraction of the edges in S may become useless for an augmentation if both its endpoints get matched in M_0 by the end of the first pass. Our main result is a two-pass algorithm (described in Section 5) that differs in two ways from the former approach. Firstly, in the first pass, we only find a maximal matching M_0 so that in the second pass, where we maintain a set S of support edges, S would not contain "useless" edges. Secondly, any augmentation in our algorithm happens immediately when an edge arrives if it forms an augmenting path of length 3 with edges in M_0 and S.

Our results

In light of the discussion so far, one way to evaluate an algorithm is how much advantage it gains over the (1/2)-approximate maximal matching found in the first pass. We summarize

Table 1 Advantages over a maximal matching – advantage α means $(1/2 + \alpha)$-approximation.

Problem	Previous work	Advantage	Advantage in this work
Bipartite two-pass	Esfandiari et al. [11]	1/12	Not considered separately
Bipartite three-pass	Esfandiari et al. [11]	1/9.52	
Triangle-free two-pass	Not considered separately		1/16 (in Section 5)
Triangle-free three-pass			1/10 (in Appendix A)
General two-pass	Konrad et al. [20]	1/140	1/32 (in Section 5)
General three-pass	Not considered separately		1/19.753 (in Appendix B)

Table 2 Multi-pass algorithms – see Section 6.

Graph	Results	Approx	# Passes
Bipartite	Feigenbaum et al. [13]	$2/3 - \varepsilon$	$O(\log(1/\varepsilon)/\varepsilon)$
	Eggert et al.[9]	$1 - \varepsilon$	$288/\varepsilon^5$
	Ahn and Guha [1]	$1 - \varepsilon$	$O(\log\log(1/\varepsilon)/\varepsilon^2)$
Triangle free	This work (in Section 6)	$2/3 - \varepsilon$	$2/(3\varepsilon)$
General	McGregor [22] *randomized*	$1 - \varepsilon$	$O((1/\varepsilon)^{1/\varepsilon})$
	Ahn and Guha [1]	$2/3 - \varepsilon$	$O(\log(1/\varepsilon)/\varepsilon^2)$
	Ahn and Guha [1]	$1 - \varepsilon$	$O(\log n \cdot \mathrm{poly}(1/\varepsilon))$
	This work (in Section 6)	$2/3 - \varepsilon$	$4/(3\varepsilon)$

our two-pass and three-pass results in Table 1 and multi-pass results in Table 2. We stress that we are able to bound the number of passes *precisely*, without big-O notation. For general graphs, our multi-pass algorithm improves the best known *deterministic* algorithms in terms of number of passes – see the third multi-row of Table 2. We note that our multi-pass algorithm is not just a repetition of the second pass of our two-pass algorithm. Such a repetition will give an asymptotically worse number of passes (see, for example, the multi-pass algorithm due to Feigenbaum et al. [13]; the first row of Table 2). We carefully choose the parameters for each pass to get the required number of passes. Also note that Table 1 shows *advantages* over a maximal matching – an algorithm is said to have advantage α if it is a $(1/2 + \alpha)$-approximation algorithm (because a maximal matching is $(1/2)$-approximate).

Note of independent work

The work of Esfandiari et al. [11] who claim better approximation ratios for *bipartite graphs* in two passes and three passes is independent and almost concurrent. Our work differs in several aspects. We consider triangle-free graphs (superset of bipartite graphs) and general graphs, and we additionally consider multi-pass algorithms. Also, their algorithm has a post-processing step that uses time $O(\sqrt{n} \cdot |E|)$, whereas our algorithms can be implemented to have $O(1)$ update time per edge. One further detail about this appears in Appendix D.

1.1 Related Work

Karp, Vazirani, and Vazirani [18] gave the celebrated $(1 - 1/e)$-competitive randomized online algorithm for bipartite graphs in the vertex arrival setting. Goel et al. [14] gave the first one-pass deterministic algorithm with the same approximation ratio, i.e., $1 - 1/e$, in the semi-streaming model in the vertex arrival setting. For the rest of this section, results involving ε hold for any $\varepsilon > 0$. As mentioned earlier, Goel, Kapralov, and Khanna [14] proved nonexistence of one-pass $(2/3+\varepsilon)$-approximation semi-streaming algorithms, which was extended to $(1-1/e+\varepsilon)$-approximation algorithms by Kapralov [16]. On the algorithms side, nothing better than outputting a maximal matching, which is $(1/2)$-approximate, is known. Closing this gap is considered an outstanding open problem in the streaming community [21].

On the multi-pass front, in the semi-streaming model, Feigenbaum et al. [13] gave a $(2/3-\varepsilon)$-approximation algorithm for bipartite graphs that uses $O(\log(1/\varepsilon)/\varepsilon)$ passes; McGregor [22] improved it to give a $(1-\varepsilon)$-approximation algorithm for general graphs that uses $O((1/\varepsilon)^{1/\varepsilon})$ passes. For bipartite graphs, this was again improved by Eggert et al. [9] who gave a $(1-\varepsilon)$-approximation $O((1/\varepsilon)^5)$-pass algorithm. Ahn and Guha [1] gave a linear-programming based $(1-\varepsilon)$-approximation $O(\log\log(1/\varepsilon)/\varepsilon^2)$-pass algorithm for bipartite graphs. For general graphs, their $(1-\varepsilon)$-approximation algorithm uses number of passes proportional to $\log n$, so it is worse than that of McGregor [22].

For the problem of one-pass weighted matching, there is a line of work starting with Feigenbaum et al. [13] giving a 6-approximation semi-streaming algorithm. Subsequent results improved this approximation ratio: see McGregor [22], Zelke [24], Epstein et al. [10], Crouch and Stubbs [8], Grigorescu et al.[15], and most recently in a breakthrough, giving a $(2+\varepsilon)$-approximation semi-streaming algorithm, Paz and Schwartzman [23]. The multi-pass version of the problem was considered first by McGregor [22], then by Ahn and Guha [1]. Chakrabarti and Kale [5] and Chekuri et al. [6] consider a more general version of the matching problem where a submodular function is defined on the edges of the input graph.

The problem of estimating the *size* of a maximum matching (instead of outputting the actual matching) has also been considered. We mention Kapralov et al. [17], Esfandiari et al. [12], Bury and Schwiegelshohn [4], and Assadi et al. [2].

In the dynamic streams, edges of the input graph can be removed as well. The works of Konrad [19], Assadi et al. [3], and Chitnis et al. [7] consider the maximum matching problem in dynamic streams.

1.2 Organization of the Paper

After setting up notation in Section 2, we give a tight analysis of the three-pass algorithm for bipartite graphs by Konrad et al. [20] in Section 3. In Section 4, we see our simple two-pass algorithm for triangle-free graphs. Then in Section 5, we see our main result – the improved two-pass algorithm, and then we see the multi-pass algorithm in Section 6. The results that are not covered in the main sections are covered in the appendix.

2 Preliminaries

We work on graph *streams*. The input is a sequence of edges (stream) of a graph $G=(V,E)$, where V is the set of vertices and E is the set of edges; a bipartite graph is denoted as $G=(A,B,E)$. A streaming algorithm may go over the stream a few times (multi-pass) and use space $O(n\operatorname{polylog} n)$, where $n=|V|$. In this paper, we give algorithms that make two, three, or a few more passes over the input graph stream. A matching M is a subset of edges such that each vertex has at most one edge in M incident to it. The maximum cardinality matching problem, or maximum matching, for short, is to find a largest matching in the given graph. Our goal is to design streaming algorithms for maximum matching.

For a subset F of edges and a subset U of vertices, we denote by $U(F)\subseteq U$ the set of vertices in U that have an edge in F incident on them. Conversely, we denote by $F(U)\subseteq F$ the set of edges in F that have an endpoint in U. For a subset F of edges and a vertex

$v \in V(F)$, we denote by $N_F(v)$ the set of v's neighbors in the graph $(V(F), F)$, and we define $\deg_F(v) := |N_F(v)|$.

In the first pass, our algorithms compute a *maximal* matching which we denote by M_0. We use M^* to indicate a matching of maximum cardinality. Assume that M_0 and M^* are given. For $i \in \{3, 5, 7, \ldots\}$, a connected component of $M_0 \cup M^*$ that is a path of length i is called an i-augmenting path (nonaugmenting otherwise). We say that an edge in M_0 is 3-augmentable if it belongs to a 3-augmenting path, otherwise we say that it is non-3-augmentable.

▶ **Lemma 1** (Lemma 1 in [20]). *Let $\alpha \geqslant 0$, M_0 be a maximal matching in G, and M^* be a maximum matching in G such that $|M_0| \leqslant (1/2+\alpha)|M^*|$. Then the number of 3-augmentable edges in M_0 is at least $(1/2 - 3\alpha)|M^*|$, and the number of non-3-augmentable edges in M_0 is at most $4\alpha|M^*|$.*

Proof. Let the number of 3-augmentable edges in M_0 be k. For each 3-augmentable edge in M_0, there are two edges in M^* incident on it. Also, each non-3-augmentable edge in M_0 lies in a connected component of $M_0 \cup M^*$ in which the ratio of the number of M^*-edges to the number of M_0-edges is at most $3/2$. Hence,

$$\begin{aligned} |M^*| &\leqslant 2k + \frac{3}{2}(|M_0| - k) && \text{since \# non-3-augmentable edges} = |M_0| - k\,, \\ &\leqslant 2k + \frac{3}{2}\left(\left(\frac{1}{2}+\alpha\right)|M^*| - k\right) && \text{because } |M_0| \leqslant (1/2+\alpha)|M^*|\,, \\ &= \frac{1}{2}k + \left(\frac{3}{4} + \frac{3}{2}\alpha\right)|M^*|\,, \end{aligned}$$

which, after simplification, gives $k \geqslant (1/2 - 3\alpha)|M^*|$. And the number of non-3-augmentable edges in M_0 is $|M_0| - k \leqslant |M_0| - (1/2-3\alpha)|M^*| \leqslant (1/2+\alpha-1/2+3\alpha)|M^*| = 4\alpha|M^*|$. ◀

We make the following simple, yet crucial, observation.

▶ **Observation 2.** *Let M_0 be a maximal matching. Then $V(M_0)$ is a vertex cover, and there is no edge between any two vertices in $V \setminus V(M_0)$. Therefore, even if the input graph is not a bipartite graph, the set of edges incident on $V \setminus V(M_0)$, i.e., $E(V \setminus V(M_0))$ give rise to a bipartite graph with bipartition $(V \setminus V(M_0), V(M_0))$.*

For all the algorithms in this paper, it can be verified that their space complexity is $O(n \log n)$ and update time per edge is $O(1)$. We also ignore floors and ceilings for the sake of exposition.

3 Analyzing the Three Pass Algorithm for Bipartite Graphs

We analyze the three-pass algorithm for bipartite graphs given by Konrad et al. [20], i.e., Algorithm 1 by considering the distribution of lengths of augmenting paths. We also give a tight example.

▶ **Theorem 3.** *Algorithm 1 is a three-pass, semi-streaming, $(1/2 + 1/10)$-approximation algorithm for maximum matching in bipartite graphs.*

Proof. Without loss of generality, let M^* be a maximum matching such that all nonaugmenting connected components of $M_0 \cup M^*$ are single edges. For $i = \{3, 5, 7, \ldots\}$, let k_i denote the number of i-augmenting paths in $M_0 \cup M^*$, and let $k = |M_0 \cap M^*|$. Then

$$|M_0| = k + \sum_i \frac{i-1}{2}k_i \quad \text{and} \quad |M^*| = k + \sum_i \frac{i+1}{2}k_i\,. \tag{1}$$

Algorithm 1 Three-pass algorithm for bipartite graphs due to Konrad et al. [20]

```
1: In the first pass, find a maximal matching $M_0$.
2: In the second pass, find a maximal matching
   - $M_A$ in $F_2 := \{ab : a \in A(M_0), b \in B \setminus B(M_0)\}$ (see Figure 1).
3: In the third pass, find a maximal matching
   - $M_B$ in $F_3 := \{ab : a \in A \setminus A(M_0)$ and $\exists a' \in A(M_A)$ such that $a'b \in M_0\}$.
4: Augment $M_0$ using edges in $M_A$ and $M_B$ and return the resulting matching $M$.
```

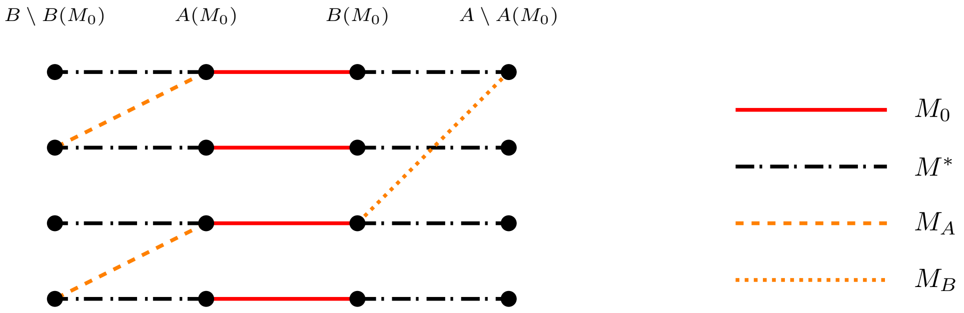


Figure 1 Example: state of variables in an execution of Algorithm 1.

Consider an i-augmenting path $b_1a_1b_2a_2b_3 \cdots b_{(i+1)/2}a_{(i+1)/2}$ in $M_0 \cup M^*$, where for each j, we have $a_j \in A$ and $b_j \in B$. We call the vertex $a_{(i-1)/2}$ a good vertex, because an edge in M_A incident to $a_{(i-1)/2}$ can potentially be augmented using the edge $b_{(i+1)/2}a_{(i+1)/2}$. To elaborate, consider the set of all edges in M_A incident on good vertices; call it M'_A. Consider the set of edges of the type $b_{(i+1)/2}a_{(i+1)/2}$ from each i-augmenting path; call it M_F. Note that M_F is a matching. Then we can augment M_0 using M'_A and M_F by as much as $|M'_A|$.

There is a matching of size $\sum_i k_i$ in F_2 formed by edges of the type b_1a_1 from each i-augmenting path. Since M_A is maximal in F_2, we have $|M_A| \geqslant (\sum_i k_i)/2$. Now, the number of good vertices is $\sum_i k_i$; therefore, the number of bad (i.e., not good) vertices is $|M_0| - \sum_i k_i$. So the number of edges in M_A incident on good vertices (see Figure 2)

$$|M'_A| \geqslant \frac{\sum_i k_i}{2} - \left(|M_0| - \sum_i k_i\right) = \frac{3}{2}\sum_i k_i - |M_0| \,.$$

Let $B_G := \{b \in B : \exists a \in A(M'_A)$ such that $ab \in M_0\}$. Let $M'_F \subseteq M_F$ be defined as $M'_F := \{ba \in M_F : b \in B_G\}$. Then we know that $|M'_F| = |M'_A|$ and $M'_F \subseteq M_F \subseteq F_3$. Since we select a maximal matching in F_3 in the third pass,

$$|M_B| \geqslant \frac{|M'_F|}{2} = \frac{|M'_A|}{2} = \frac{3}{4}\sum_i k_i - \frac{|M_0|}{2} \,. \tag{2}$$

So the output size

$$\begin{aligned} |M| &= |M_0| + |M_B| \\ &\geqslant |M_0| + \frac{3}{4}\sum_i k_i - \frac{|M_0|}{2} && \text{by (1) and (2),} \\ &= \frac{|M_0|}{2} + \frac{3}{4}(|M^*| - |M_0|) && \text{by (1), } \sum_i k_i = |M^*| - |M_0| \,, \end{aligned}$$

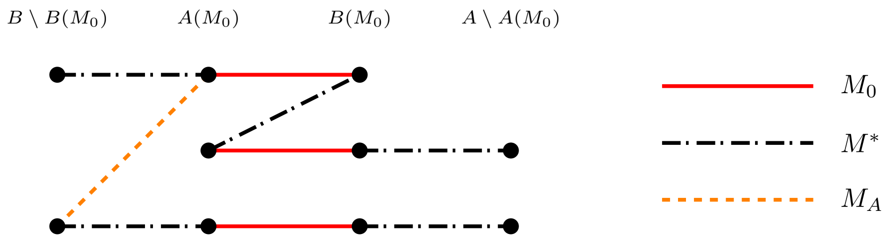


Figure 2 Tight example for Algorithm 1: M_A has only one edge that lands on a bad vertex and cannot be augmented in the third pass. So $|M| = |M_0| = 3$ and $|M^*| = 5$.

Algorithm 2 Two-pass algorithm for triangle-free graphs

```
In the first pass: $M_0 \leftarrow$ maximal matching
In the second pass: $S \leftarrow \textsc{Semi}(\lambda, V(M_0), V \setminus V(M_0))$ (see Figure 3).
After the second pass, augment $M_0$ greedily using edges in $S$ to get $M$; output $M$.
function $\textsc{Semi}(\lambda, X, Y)$                    ▷ based on Algorithm 7 in Konrad et al. [20]
    $S \leftarrow \emptyset$
    foreach edge $xy$ such that $x \in X$, $y \in Y$ do
        if $\deg_S(x) = 0$ and $\deg_S(y) \leqslant \lambda - 1$ then
            $S \leftarrow S \cup \{xy\}$
```

i.e., $|M| \geqslant 3|M^*|/4 - |M_0|/4$, but we also have $|M| \geqslant |M_0|$, hence

$$|M| \geqslant \max \left\{ |M_0|, \frac{3}{4}|M^*| - \frac{1}{4}|M_0| \right\} .$$

So the bound is minimized when $|M_0| = 3|M^*|/4 - |M_0|/4 = 3|M^*|/5 = (1/2+1/10)|M^*|$. ◀

As we can see in the proof above, the worst case happens when $|M| = |M_0| = 3|M^*|/5$. Setting $k_3 = k_5 \geqslant 1$, $k = 0$, and $k_i = 0$ for $i > 5$ gives us the tight example shown in Figure 2.

4 A Simple Two Pass Algorithm for Triangle Free Graphs

Before seeing our main result, we see a simple two pass algorithm for triangle-free graphs. The function $\textsc{Semi}()$ in Algorithm 2 greedily computes a subset of edges such that each vertex in X has degree at most one and each vertex in Y has degree at most λ; we call such a subset a (λ, X, Y)-semi-matching (Konrad et al. [20] call this a λ-bounded semi-matching). In Algorithm 2, we find a maximal matching M_0 in the first pass, and, in the second pass, we find a $(\lambda, V(M_0), V \setminus V(M_0))$-semi-matching S. After the second pass, we greedily augment edges in M_0 one by one using edges in S.

▶ **Theorem 4.** *Algorithm 2 is a two-pass, semi-streaming,* $(1/2 + 1/20)$*-approximation algorithm for maximum matching in triangle-free graphs.*

Proof. As in the proof of Theorem 3, let M^* be a maximum matching such that all nonaugmenting connected components of $M_0 \cup M^*$ are single edges. For $i = \{3, 5, 7, \ldots\}$, let k_i denote the number of i-augmenting paths in $M_0 \cup M^*$, and let k denote the number of edges in $M^* \cap M_0$.

Consider an i-augmenting path $x_1 y_1 x_2 y_2 x_3 \cdots x_{(i+1)/2} y_{(i+1)/2}$ in $M_0 \cup M^*$. We call the vertices $y_1 \in V(M_0)$ and $x_{(i+1)/2} \in V(M_0)$ good vertices, because the edges $x_1 y_1 \in M^*$ and

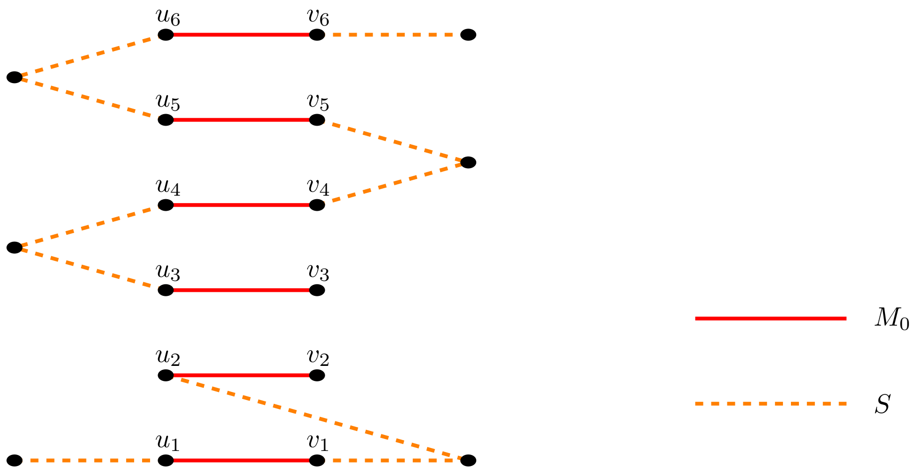


Figure 3 Example showing M_0 and S at the end of the second pass of Algorithm 2 with $\lambda = 2$. When we greedily augment M_0 after the second pass, we may choose to augment u_5v_5 and lose two possible augmentations of edges u_4v_4 and u_6v_6.

$x_{(i+1)/2}y_{(i+1)/2} \in M^*$ can potentially be added to S by our algorithm. Denote by V_G the set of good vertices and by $V_B := V(M_0) \setminus V_G$ the set of bad vertices. Then $|V_G| = 2\sum_i k_i$. Note that $V_G \cap V_B = \emptyset$ and $V_G \cup V_B = V(M_0)$ by definition.

Let $V_{\text{NC}} := V_G \setminus V(S)$ be the set of good vertices *not covered* by S. An edge $uv \in M^*$ with $u \in V \setminus V(M_0)$ and $v \in V_{\text{NC}}$ was not added to S, because $\deg_S(u) = \lambda$. Hence

$$\lambda|V_{\text{NC}}| \leqslant |V(M_0)| - |V_{\text{NC}}| \quad \text{i.e.,} \quad |V_{\text{NC}}| \leqslant \frac{2}{\lambda+1}|M_0|\,, \tag{3}$$

because at most $|V(M_0)| - |V_{\text{NC}}|$ vertices in $V(M_0)$ are covered by S. Now,

$$\begin{aligned}|V(M_0) \setminus V(S)| &= |V_G \setminus V(S)| + |V_B \setminus V(S)| && \because V_G \cap V_B = \emptyset \text{ and } V_G \cup V_B = V(M_0),\\ &\leqslant |V_{\text{NC}}| + |V_B| && \because V_{\text{NC}} = V_G \setminus V(S), |V_B \setminus V(S)| \leqslant |V_B|,\\ &\leqslant \frac{2}{\lambda+1}|M_0| + |V(M_0)| - |V_G| && \text{by (3) and} \because |V_B| = |V(M_0)| - |V_G|,\\ &= \frac{2}{\lambda+1}|M_0| + |V(M_0)| - 2\sum_i k_i && \text{because } |V_G| = 2\sum_i k_i\,.\end{aligned}$$

Using $|V(M_0)| = |V(M_0) \setminus V(S)| + |V(M_0) \cap V(S)|$ and the above, we get

$$\begin{aligned}|V(M_0) \cap V(S)| &\geqslant |V(M_0)| - \left(\frac{2}{\lambda+1}|M_0| + |V(M_0)| - 2\sum_i k_i\right)\\ &= 2\left(\sum_i k_i - \frac{1}{\lambda+1}|M_0|\right)\,. \end{aligned} \tag{4}$$

We observe that at most $|M_0|$ vertices in $V(M_0)$ (one endpoint of each edge) can be covered by S without having both endpoints of an edge in M_0 covered. Hence, at least $|V(M_0) \cap V(S)| - |M_0|$ edges in M_0 have both their endpoints covered by S, which, by (4), is at least

$$2\left(\sum_i k_i - \frac{1}{\lambda+1}|M_0|\right) - |M_0| = 2\sum_i k_i - \frac{\lambda+3}{\lambda+1}|M_0|\,. \tag{5}$$

After the second pass, when we greedily augment an edge from the above edges, i.e., edges whose both endpoints are covered by S, we may potentially lose $2(\lambda-1)$ other augmentations (see Figure 3). To see this, consider $uv \in M_0$ such that $u, v \in V(S)$ and $au \in S$ and $vb \in S$. The graph is triangle free, so we know that $a \neq b$, and we *can* augment M_0 using the 3-augmenting path $auvb$; but we may lose at most $\lambda - 1$ edges incident to a in S and at most $\lambda - 1$ edges incident to b in S. Therefore the number of augmentations c we get after the second pass is at least $1/(2\lambda-1)$ times the right hand side of (5), i.e.,

$$c \geqslant \frac{2}{2\lambda-1}\sum_i k_i - \frac{\lambda+3}{(2\lambda-1)(\lambda+1)}|M_0| .$$

So the output size $|M| = |M_0| + c$, and using the above bound on c and simplifying we get:

$$|M| \geqslant \frac{2}{2\lambda-1}\sum_i k_i + \frac{2(\lambda^2-2)}{(2\lambda-1)(\lambda+1)}|M_0| ;$$

substituting $\sum_i k_i = |M^*| - |M_0|$, by (1), in the above,

$$|M| \geqslant \frac{2}{2\lambda-1}|M^*| + \frac{2(\lambda^2-\lambda-3)}{(2\lambda-1)(\lambda+1)}|M_0| .$$

Using $\lambda = 3$ and the fact that M_0 is 2-approximate, we get

$$|M| \geqslant \frac{2}{5}|M^*| + \frac{3}{10}|M_0| \geqslant \frac{2}{5}|M^*| + \frac{3}{20}|M^*| = \frac{11}{20}|M^*| = \left(\frac{1}{2}+\frac{1}{20}\right)|M^*| . \qquad \blacktriangleleft$$

5 Improved Two Pass Algorithm

We present our main result that is a two pass algorithm in this section. In the first pass, we find a maximal matching M_0. In the second pass, we maintain a set S of support edges xy, such that $x \in V \setminus V(M_0)$, $y \in V(M_0)$, and $\deg_S(y) \leqslant \lambda_{\mathrm{M}}$ and $\deg_S(x) \leqslant \lambda_{\mathrm{U}}$, where $\lambda_{\mathrm{M}} \geqslant 1$ and $\lambda_{\mathrm{U}} \geqslant 1$ are parameters denoting maximum degree allowed in S for matched and unmatched vertices (with respect to M_0), respectively. Whenever a new edge forms a 3-augmenting path with an edge in M_0 and an edge in S, we augment. We store the vertices involved in a 3-augmentation in the variable I. We ignore a new edge if it is incident to a vertex in I. Unused support edges that are incident to a vertex in I become "useless"; hence to address this, we store the endpoints of M_0 edges that share an endpoint with such useless edges in the variable I_B, and we ignore a new edge if it is incident to a vertex in I_B. Algorithm 3 gives a formal description.

Setting up a charging scheme to lower bound the number of augmentations

We first lay the groundwork and give a charging scheme.

▶ **Observation 5.** *For general graphs (that are possibly not triangle-free), we need to set* $\lambda_{\mathrm{M}} \geqslant 2$.

To see why, suppose $\lambda_{\mathrm{M}} = 1$. Let uv be a 3-augmentable edge in M_0. Then, for the edge uv, we might end up storing the edges ub and vb in S, and the edge uv would not get augmented. If $\lambda_{\mathrm{M}} \geqslant 2$, and we store at least λ_{M} edges incident to u, then an edge incident to v will

Algorithm 3 Improved two-pass algorithm: input graph G

```
In the first pass, find a maximal matching $M_0$.
if $G$ is triangle-free then
    Return Improve-Matching($M_0, 2, 1$)
else
    Return Improve-Matching($M_0, 4, 2$)
function Improve-Matching($M_0, \lambda_U, \lambda_M$)
    $M \leftarrow M_0$, $S \leftarrow \emptyset$, $I \leftarrow \emptyset$ and $I_B \leftarrow \emptyset$
    foreach edge $xy$ in the stream do
        if $x$ or $y \in I \cup I_B$ then
            Continue, i.e., ignore $xy$.
        else if $x \in V(M_0)$ and $y \in V(M_0)$ then
            Continue, i.e., ignore $xy$.
        else if there exist $v$ and $b$ such that $yv \in M_0$ and $vb \in S$ then
            $M \leftarrow M \setminus \{yv\} \cup \{xy, vb\}$                ▷ a 3-augmentation
            Let $I_x \leftarrow \{u_x, v_x : xu_x \in S$ and $u_x v_x \in M_0\}$.
            Let $I_b \leftarrow \{u_b, v_b : u_b v_b \in M_0$ and $v_b b \in S\}$.
            Then $I \leftarrow I \cup \{x, y, v, b\}$ and $I_B \leftarrow I_B \cup I_x \cup I_b$.
        else
            Without loss of generality, assume that $x \in V \setminus V(M_0)$ and $y \in V(M_0)$.
            if $\deg_S(x) < \lambda_U$ and $\deg_S(y) < \lambda_M$ then          ▷ See Figure 4.
                $S \leftarrow S \cup \{xy\}$   ▷ Note: Once an edge is added to $S$, it is never removed
    from it.
    Return $M$.
```

not form a triangle with at least one of those and uv would get augmented. So, for general graphs, we need to set $\lambda_M \geqslant 2$.

Let $|M_0| = (1/2 + \alpha)|M^*|$. For a 3-augmentable edge $uv \in M_0$, let $auvb$ be the 3-augmenting path such that $au, vb \in M^*$. Without loss of generality, assume that au arrived before vb. Then we make the following observation.

▶ **Observation 6.** *When au arrived, it may not be added to S for one of the following reasons:*

- *The vertex a was already matched.*
- *There were λ_M edges incident to u in S.*
- *There were λ_U edges incident to a in S.*

We call some edges in M_0 *good*, some *partially* good, and some *bad*. An edge is good if it got augmented. An edge $uv \in M_0$ is bad if it is 3-augmentable, not good, and vertex a or b had λ_U edges incident to them in S when edge au or vb arrived. An edge $uv \in M_0$ is partially good if it is 3-augmentable, but neither good nor bad ("partially" good because, as we will see later, we can hold some good edge $u'v' \in M_0$ responsible for uv not getting augmented). Note that all 3-augmentable edges get some label according to our labeling. We require the following lemma to describe the charging scheme.

▶ **Lemma 7.** *Suppose au was not added to S because there were already λ_M edges incident to u in S. If, later, uv did not get augmented when vb arrived, then*

- *b was already matched via augmenting path $a''u''v''b$, or*
- *there exists $a'u \in S$ and $u'v' \in M_0$ such that a' was matched via augmenting path $a'u'v'b'$.*

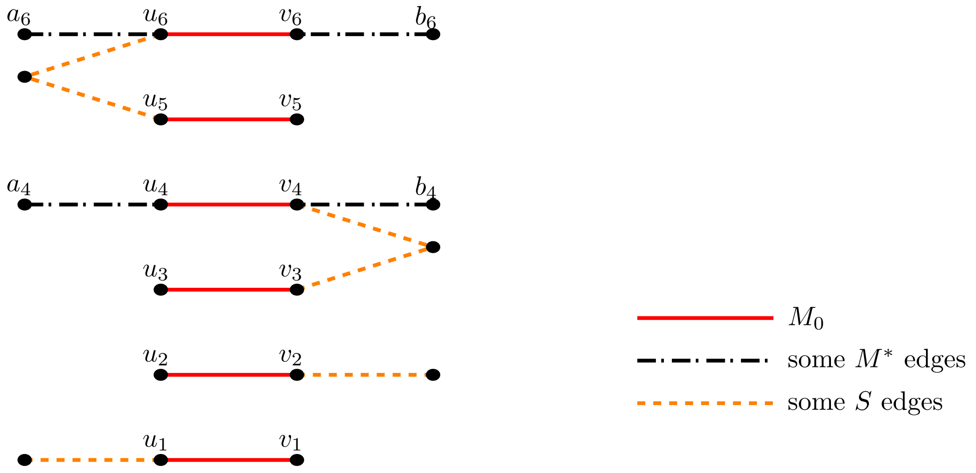


Figure 4 Example showing M_0 and some of the edges in M^* and S during the second pass of Algorithm 3 for triangle-free graphs with $\lambda_{\mathrm{U}} = 2$ and $\lambda_{\mathrm{M}} = 1$. At most one of u_i and v_i can have positive degree in S, because we would rather augment u_iv_i instead of adding the latter edge to S. By our convention, a_4u_4 arrived before v_4b_4, and a_6u_6 arrived before v_6b_6. Since a_4u_4 was not added to S, we have $\deg_S(a_4) = \lambda_{\mathrm{U}}$ (S edges incident to a_4 are not shown).

Proof. When au arrived, $|N_S(u)| \geqslant \lambda_{\mathrm{M}}$. If b was unmatched when vb arrived, then some $a' \in N_S(u) \setminus \{b\}$ must have been matched, otherwise we would have augmented uv. Now for triangle-free graphs $b \notin N_S(u)$, so $|N_S(u) \setminus \{b\}| = |N_S(u)| \geqslant 1$, and for general graphs, by Observation 5, $\lambda_{\mathrm{M}} \geqslant 2$, so $|N_S(u) \setminus \{b\}| \geqslant \lambda_{\mathrm{M}} - 1 \geqslant 1$. ◀

Charging Scheme

As alluded to earlier, we charge a partially good edge to some good edge. Recall that for a 3-augmentable edge $uv \in M_0$, we denote by $au, vb \in M^*$ the edges that form the 3-augmenting path with uv such that au arrived before vb. We use Observation 6 and consider the following cases. See Figure 5.

- Suppose au was not added to S because a was already matched. Then, let $u'v' \in M_0$ was augmented using $au'v'b'$. If $\deg_S(a) \leqslant \lambda_{\mathrm{U}} - 1$, then we charge uv to $u'v'$. Otherwise, uv is bad.
- Suppose au was not added to S because $\deg_S(u) = \lambda_{\mathrm{M}}$. Then we use Lemma 7. We either charge uv to $u'v'$, or if $\deg_S(b) \leqslant \lambda_{\mathrm{U}} - 1$, then we charge uv to $u''v''$. Otherwise, uv is bad.
- Suppose au was not added to S because $\deg_S(a) = \lambda_{\mathrm{U}}$, then uv is bad.
- Otherwise, au was added to S, but uv did not get augmented when vb arrived. Then:
  - Either there exists $a' \in N_S(u)$ that was matched via augmenting path $a'u'v'b'$ (note that a' may be same as a), then we charge uv to $u'v'$;
  - or b was already matched via augmenting path $a''u''v''b$, and vb was ignored; in this case, if $\deg_S(b) \leqslant \lambda_{\mathrm{U}} - 1$, then we charge uv to $u''v''$, otherwise, uv is bad.

We now bound the number of bad edges in M_0 from above.

▶ **Lemma 8.** *The number of bad edges is at most* $\lambda_{\mathrm{M}}|M_0|/\lambda_{\mathrm{U}}$.

Proof. We claim that for any $uv \in M_0$, $\deg_S(u) + \deg_S(v) \leqslant \lambda_{\mathrm{M}}$, hence $|S| \leqslant \lambda_{\mathrm{M}}|M_0|$. A short argument is that the $(\lambda_{\mathrm{M}} + 1)$th edge would cause an augmentation and will not

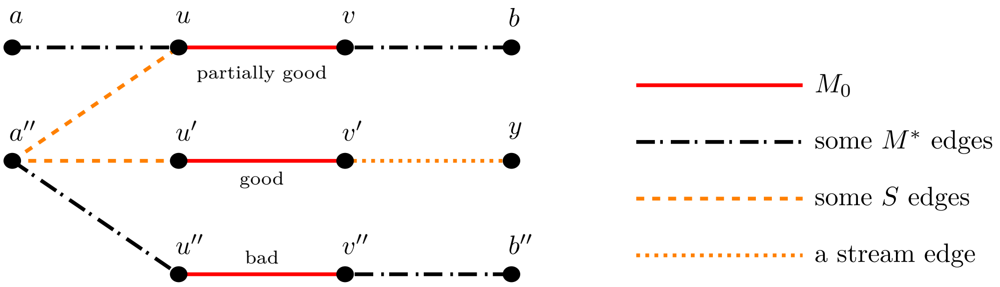


Figure 5 Example showing a good edge, a bad edge, and a partially good edge. We use parameters $\lambda_U = 2$ and $\lambda_M = 1$, so we are in the triangle-free case. The edge $u'v'$ is not 3-augmentable but was augmented using $a''u'v'y$, so $u'v'$ is a good edge. The edge $u''v''$ is a 3-augmentable edge that was not augmented and when $a''u''$ arrived, $\deg_S(a'') = 2$, so $u''v''$ is a bad edge. For uv, we did not take au in S, because $\deg_S(u) = 1$, so uv is a partially good edge, and we can charge uv to $u'v'$ using Lemma 7.

be added to S. Let us assume the claim. By the definition of a bad edge, λ_U edges in S are "responsible" for one bad edge in M_0. Also, an edge au' (or $v''b$, resp.) in S can be responsible for at most one bad edge that can only be uv if $au \notin S$ (or if $vb \notin S$, resp.; considering the 3-augmenting path $auvb$). Hence, the total number of bad edges is at most $|S|/\lambda_U \leqslant \lambda_M|M_0|/\lambda_U$. Now we prove the claim.

We first prove for triangle-free graphs by contradiction. Let $\deg_S(u) + \deg_S(v) > \lambda_M$, and let $vy \in S$ be the $(\lambda_M + 1)$th edge incident to one of u and v that was added to S. Since $\lambda_M \geqslant 1$ and $\deg_S(v) \leqslant \lambda_M$, we have $\deg_S(u) \geqslant 1$, i.e. $N_S(u) \neq \emptyset$. Now when vy arrived:

- the vertex y was unmatched, otherwise vy would not be added to S;
- no vertex $x \in N_S(u)$ was matched, otherwise $u, v \in I_B$, and vy would not be added to S.

The above implies that when vy arrived, due to some $x \in N_S(u)$ the if condition on Line 14 became true, and we augmented uv via $xuvy$ instead of adding vy to S. This is a contradiction.

For general graphs, we argue by contradiction slightly informally for the sake of brevity. By Observation 5, for general graphs, $\lambda_M \geqslant 2$. Let $\deg_S(u) + \deg_S(v) > \lambda_M \geqslant 2$. Let vy be the second edge incident to one of u and v that was added to S; the first edge can be xu or vy'.

Suppose xu was the first edge. If $x \neq y$, then we would have augmented uv via $xuvy$ instead of adding vy to S – a contradiction. If $x = y$, then after vy was processed, $N_S(u) = N_S(v) = \{y\}$, and a third edge incident to one of u and v would not be added to S, because it would have formed a 3-augmenting path with either yu or vy, resulting in a contradiction that $\deg_S(u) + \deg_S(v) = 2$.

Otherwise, suppose vy' was the first edge; then $N_S(v) = \{y, y'\}$ after vy was processed. Since eventually $\deg_S(u) + \deg_S(v) \geqslant \lambda_M + 1 \geqslant 3$ and $\deg_S(u), \deg_S(v) \leqslant \lambda_M$, we would eventually have $\deg_S(u) \geqslant 1$, so let $xu \in S$. When xu arrived, it would have formed an 3-augmenting path with either vy or vy' (here, taking care of the fact that one of y and y' can be same as x), resulting in a contradiction that xu was not added to S.

Thus, we get the claim and complete the proof. ◀

As a consequence, we get the following.

▶ **Observation 9.** *In any call to* Improve-Matching(), *we need to set* $\lambda_U > \lambda_M$, *i.e.,* $\lambda_U \geqslant 2$.

To see why, suppose $\lambda_{\mathrm{U}} \leqslant \lambda_{\mathrm{M}}$. Then by Lemma 8, potentially all 3-augmentable edges in M_0 could become bad edges.

Recall that a 3-augmentable edge is good, partially good, or bad; so by Lemmas 1 and 8,

$$\begin{aligned} \#\text{ good or partially good edges} &\geqslant \left(\frac{1}{2} - 3\alpha\right)|M^*| - \frac{\lambda_{\mathrm{M}}|M_0|}{\lambda_{\mathrm{U}}} \\ &= \left(\frac{1}{2} - 3\alpha\right)|M^*| - \frac{\lambda_{\mathrm{M}}}{\lambda_{\mathrm{U}}}\left(\frac{1}{2}+\alpha\right)|M^*| \\ &= \left(\frac{\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}}}{2\lambda_{\mathrm{U}}} - \left(\frac{3\lambda_{\mathrm{U}} + \lambda_{\mathrm{M}}}{\lambda_{\mathrm{U}}}\right)\alpha\right)|M^*|\,. \qquad (6) \end{aligned}$$

In the following lemma, we bound the number of partially good edges in M_0 that are charged to one good edge.

▶ **Lemma 10.** *At most $2\lambda_{\mathrm{U}} - 1$ partially good edges in M_0 are charged to one good edge in M_0.*

Proof. Suppose $uv \in M_0$ was augmented by edges xu and vy such that xu arrived before vy, then $xu \in S$. Now $|N_S(x)|, |N_S(y)| \leqslant \lambda_{\mathrm{U}}$. Since $xu \in S$, we have $|N_S(x) \setminus \{u\}| \leqslant \lambda_{\mathrm{U}} - 1$. Let $B := (N_S(x) \setminus \{u\}) \cup N_S(y)$, then $|B| \leqslant 2\lambda_{\mathrm{U}} - 1$. Now, the set of partially good edges that are charged to uv is a subset of $M_0(B)$. Observing that $|M_0(B)| \leqslant |B| \leqslant 2\lambda_{\mathrm{U}} - 1$ finishes the proof. ◀

The following lemma characterizes the improvement given by Improve-Matching().

▶ **Lemma 11.** *Let $|M_0| = (1/2+\alpha)|M^*|$ and $M = \textsc{Improve-Matching}(M_0, \lambda_{\mathrm{U}}, \lambda_{\mathrm{M}})$, then*

$$|M| \geqslant \left(\frac{1}{2} + \frac{\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}}}{4\lambda_{\mathrm{U}}^2} + \left(1 - \frac{3\lambda_{\mathrm{U}} + \lambda_{\mathrm{M}}}{2\lambda_{\mathrm{U}}^2}\right)\alpha\right)|M^*| \geqslant \left(\frac{1}{2} + \frac{\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}}}{4\lambda_{\mathrm{U}}^2}\right)|M^*|\,.$$

Proof. By (6) and Lemma 10, the total number of augmentations during one call to Improve-Matching() is at least

$$\frac{1}{2\lambda_{\mathrm{U}}}\left(\frac{\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}}}{2\lambda_{\mathrm{U}}} - \left(\frac{3\lambda_{\mathrm{U}} + \lambda_{\mathrm{M}}}{\lambda_{\mathrm{U}}}\right)\alpha\right)|M^*| = \left(\frac{\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}}}{4\lambda_{\mathrm{U}}^2} - \left(\frac{3\lambda_{\mathrm{U}} + \lambda_{\mathrm{M}}}{2\lambda_{\mathrm{U}}^2}\right)\alpha\right)|M^*|\,.$$

Hence, we get the following bound on the size of the output matching M:

$$\begin{aligned} |M| &\geqslant |M_0| + \left(\frac{\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}}}{4\lambda_{\mathrm{U}}^2} - \frac{3\lambda_{\mathrm{U}} + \lambda_{\mathrm{M}}}{2\lambda_{\mathrm{U}}^2}\alpha\right)|M^*| \\ &= \left(\frac{1}{2} + \frac{\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}}}{4\lambda_{\mathrm{U}}^2} + \left(1 - \frac{3\lambda_{\mathrm{U}} + \lambda_{\mathrm{M}}}{2\lambda_{\mathrm{U}}^2}\right)\alpha\right)|M^*| \quad \text{because } |M_0| = (1/2+\alpha)|M^*|\ , \\ &\geqslant \left(\frac{1}{2} + \frac{\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}}}{4\lambda_{\mathrm{U}}^2}\right)|M^*| \qquad \text{since } \lambda_{\mathrm{U}} \geqslant 2 \text{ by Observation 9}\,. \end{aligned}$$ ◀

Now we state and prove our main result.

▶ **Theorem 12.** *Algorithm 3 uses two passes and has an approximation ratio of $1/2 + 1/16$ for triangle-free graphs and an approximation ratio of $1/2 + 1/32$ for general graphs for maximum matching.*

Proof. After the second pass, the output size $|M| \geqslant (1/2 + (\lambda_{\mathrm{U}} - \lambda_{\mathrm{M}})/(4\lambda_{\mathrm{U}}^2))|M^*|$ due to Lemma 11; we use $\lambda_{\mathrm{U}} = 2$ and $\lambda_{\mathrm{M}} = 1$ for triangle-free graphs and $\lambda_{\mathrm{U}} = 4$ and $\lambda_{\mathrm{M}} = 2$ (see Observation 5) for general graphs to get the claimed approximation ratios. ◀

Algorithm 4 Multi-pass algorithm: input graph G

```
In the first pass, find a maximal matching $M_1$.
$M \leftarrow M_1$
if $G$ is triangle-free then
    for $i = 2$ to $\lceil 2/(3\varepsilon) \rceil$ do
        $M \leftarrow$ Improve-Matching$(M, i, 1)$
else
    for $i = 2$ to $\lceil 4/(3\varepsilon) \rceil$ do
        $M \leftarrow$ Improve-Matching$(M, i+1, 2)$
Return $M$.
```

6 Multi Pass Algorithm

We run the function Improve-Matching() in Algorithm 3 with increasing values of λ_{U}, and the approximation ratio converges to $1/2 + 1/6$. We note that this multi-pass algorithm is not just a repetition of the function Improve-Matching(). Such a repetition will give an asymptotically worse number of passes (see, for example, the multi-pass algorithm due to Feigenbaum et al. [13]). We carefully choose the parameter λ_{U} for each pass to get the required number of passes.

▶ **Theorem 13.** *For any $\varepsilon > 0$, Algorithm 4 is a semi-streaming $(1/2+1/6-\varepsilon)$-approximation algorithm for maximum matching that uses $2/(3\varepsilon)$ passes for triangle-free graphs and $4/(3\varepsilon)$ passes for general graphs.*

Proof. We prove the theorem for triangle-free case; the general case is similar. Let M_i be the matching computed by Algorithm 4 after ith pass, and let $p := \lceil 2/(3\varepsilon) \rceil$, so $\varepsilon \leqslant 2/(3p)$. Since M_1 is maximal, it is $(1/2)$-approximate. Let $\alpha_1 := 0$, and for $i \in \{2, 3, \dots, p\}$, let

$$\alpha_i := \frac{i-1}{4i^2} + \left(1 - \frac{3i+1}{2i^2}\right)\alpha_{i-1}$$

(see Lemma 11 with $\lambda_{\mathrm{U}} = i$ and $\lambda_{\mathrm{M}} = 1$). Then, by Lemma 11 and the logic of Algorithm 4, for $i \in [p]$, the matching M_i is $(1/2 + \alpha_i)$-approximate (by a trivial induction). Now we bound α_p by induction. We claim that for $i \in [p]$,

$$\alpha_i \geqslant \frac{1}{6} - \frac{2}{3i}\,,$$

which we prove by induction on i.

Base case: For $i = 1$, we have $1/6 - \alpha_1 = 1/6 - 0 = 1/6 \leqslant 2/(3 \cdot 1)$.

For inductive step, we want to show that

$$\frac{1}{6} - \alpha_i = \frac{1}{6} - \frac{i-1}{4i^2} - \left(1 - \frac{3i+1}{2i^2}\right)\alpha_{i-1} \leqslant \frac{2}{3i}\,,$$

which is implied by the following (using inductive hypothesis)

$$\frac{1}{6} - \frac{i-1}{4i^2} + \left(1 - \frac{3i+1}{2i^2}\right)\left(\frac{2}{3(i-1)} - \frac{1}{6}\right) \leqslant \frac{2}{3i}\,,$$

implied by

$$\frac{1}{6} - \frac{i-1}{4i^2} + \left(\frac{2i^2 - 3i - 1}{2i^2}\right)\left(\frac{4 - i + 1}{6(i-1)}\right) \leqslant \frac{2}{3i}\,,$$

multiplying both sides by $12i^2(i-1)$, we then need to show that,

$$\begin{aligned}
& & 2i^2(i-1) - 3(i-1)^2 + (2i^2 - 3i - 1)(-i+5) &\leqslant 8i(i-1)\,, \\
&\text{implied by} & 2i^3 - 2i^2 - 3(i^2 - 2i + 1) + (-2i^3 + 10i^2 + 3i^2 - 15i + i - 5) &\leqslant 8i^2 - 8i\,, \\
&\text{implied by} & 2i^3 - 5i^2 + 6i - 3 + (-2i^3 + 13i^2 - 14i - 5) &\leqslant 8i^2 - 8i\,, \\
&\text{implied by} & 8i^2 - 8i - 8 &\leqslant 8i^2 - 8i\,,
\end{aligned}$$

which is true, so we get the claim. Therefore $\alpha_p \geqslant 1/6 - 2/(3p) \geqslant 1/6 - \varepsilon$, and by our earlier observation, M_p is $(1/2 + \alpha_p)$-approximate, and this finishes the proof for triangle-free case. The proof for general case is very similar. We define $p := \lceil 4/(3\varepsilon) \rceil$ and $\alpha_1 := 0$, and for $i \in \{2, 3, \ldots, p\}$, we define

$$\alpha_i := \frac{i-1}{4(i+1)^2} + \left(1 - \frac{3(i+1)+2}{2(i+1)^2}\right)\alpha_{i-1}\,,$$

i.e., we use $\lambda_{\mathrm{U}} = i+1$ and $\lambda_{\mathrm{M}} = 2$. The corresponding claim then is that for $i \in [p]$,

$$\alpha_i \geqslant \frac{1}{6} - \frac{4}{3i}\,,$$

which can be verified by induction on i. ◀

Acknowledgements. We thank Sundar Vishwanathan and Ashish Chiplunkar for helpful discussions. The first author would like to thank his advisor Amit Chakrabarti and Andrew McGregor for helpful discussions.

References

1 Kook Jin Ahn and Sudipto Guha. Linear programming in the semi-streaming model with application to the maximum matching problem. *Inf. Comput.*, 222:59–79, January 2013. `doi:10.1016/j.ic.2012.10.006`.

2 Sepehr Assadi, Sanjeev Khanna, and Yang Li. On estimating maximum matching size in graph streams. In *Proc. 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1723–1742, 2017. `doi:10.1137/1.9781611974782.113`.

3 Sepehr Assadi, Sanjeev Khanna, Yang Li, and Grigory Yaroslavtsev. Maximum matchings in dynamic graph streams and the simultaneous communication model. In *Proc. 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1345–1364, 2016. URL: `http://dl.acm.org/citation.cfm?id=2884435.2884528`.

4 Marc Bury and Chris Schwiegelshohn. Sublinear estimation of weighted matchings in dynamic data streams. In *Proc. 23rd Annual European Symposium on Algorithms*, pages 263–274, 2015. `doi:10.1007/978-3-662-48350-3_23`.

5 Amit Chakrabarti and Sagar Kale. Submodular maximization meets streaming: matchings, matroids, and more. *Mathematical Programming*, 154(1):225–247, 2015. `doi:10.1007/s10107-015-0900-7`.

6 Chandra Chekuri, Shalmoli Gupta, and Kent Quanrud. Streaming algorithms for submodular function maximization. In *Proc. 42nd International Colloquium on Automata, Languages and Programming*, pages 318–330, 2015. `doi:10.1007/978-3-662-47672-7_26`.

7 Rajesh Chitnis, Graham Cormode, Hossein Esfandiari, MohammadTaghi Hajiaghayi, Andrew McGregor, Morteza Monemizadeh, and Sofya Vorotnikova. Kernelization via sampling with applications to finding matchings and related problems in dynamic graph streams. In *Proc. 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1326–1344, 2016. URL: `http://dl.acm.org/citation.cfm?id=2884435.2884527`.

8 Michael Crouch and Daniel M. Stubbs. Improved streaming algorithms for weighted matching, via unweighted matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*, volume 28 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 96–104. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.96.

9 Sebastian Eggert, Lasse Kliemann, Peter Munstermann, and Anand Srivastav. Bipartite matching in the semi-streaming model. *Algorithmica*, 63(1):490–508, 2012. doi:10.1007/s00453-011-9556-8.

10 Leah Epstein, Asaf Levin, Julian Mestre, and Danny Segev. Improved approximation guarantees for weighted matching in the semi-streaming model. *SIAM Journal on Discrete Mathematics*, 25(3):1251–1265, 2011. doi:10.1137/100801901.

11 H. Esfandiari, M. Hajiaghayi, and M. Monemizadeh. Finding large matchings in semi-streaming. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, pages 608–614, Dec 2016. doi:10.1109/ICDMW.2016.0092.

12 Hossein Esfandiari, Mohammad T. Hajiaghayi, Vahid Liaghat, Morteza Monemizadeh, and Krzysztof Onak. Streaming algorithms for estimating the matching size in planar graphs and beyond. In *Proc. 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1217–1233, 2015. URL: http://dl.acm.org/citation.cfm?id=2722129.2722210.

13 Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. On graph problems in a semi-streaming model. *Theor. Comput. Sci.*, 348(2):207–216, December 2005. doi:10.1016/j.tcs.2005.09.013.

14 Ashish Goel, Michael Kapralov, and Sanjeev Khanna. On the communication and streaming complexity of maximum bipartite matching. In *Proc. 23rd Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 468–485, 2012. URL: http://dl.acm.org/citation.cfm?id=2095116.2095157.

15 Elena Grigorescu, Morteza Monemizadeh, and Samson Zhou. Streaming weighted matchings: Optimal meets greedy. *CoRR*, abs/1608.01487, 2016. URL: http://arxiv.org/abs/1608.01487.

16 Michael Kapralov. Better bounds for matchings in the streaming model. In *Proc. 24th Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2013.

17 Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Approximating matching size from random streams. In *Proc. 25th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 734–751, 2014. URL: http://dl.acm.org/citation.cfm?id=2634074.2634129.

18 Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An optimal algorithm for on-line bipartite matching. In *Proc. 22nd Annual ACM Symposium on the Theory of Computing*, pages 352–358, 1990. doi:10.1145/100216.100262.

19 Christian Konrad. Maximum matching in turnstile streams. In *Proc. 23rd Annual European Symposium on Algorithms*, pages 840–852, 2015. doi:10.1007/978-3-662-48350-3_70.

20 Christian Konrad, Frédéric Magniez, and Claire Mathieu. Maximum matching in semi-streaming with few passes. *In Proc. 15th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, pages 231–242, 2012 and CoRR*, abs/1112.0184, 2014. URL: http://arxiv.org/abs/1112.0184.

21 Andrew McGregor. Problem 60: Single-pass unweighted matchings. http://sublinear.info/index.php?title=Open_Problems:60. Accessed: 2017-02-16.

22 Andrew McGregor. Finding graph matchings in data streams. In *Proc. 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 170–181, 2005. doi:10.1007/11538462_15.

23 Ami Paz and Gregory Schwartzman. A $(2+\varepsilon)$-approximation for maximum weight matching in the semi-streaming model. In *Proc. 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2153–2161, 2017. doi:10.1137/1.9781611974782.140.

24 Mariano Zelke. Weighted matching in the semi-streaming model. In *Proc. 25th International Symposium on Theoretical Aspects of Computer Science*, pages 669–680, 2008.

Algorithm 5 Three-pass algorithm for triangle-free graphs

```
In the first pass, find a maximal matching $M_0$.
In the second pass, find a maximal matching $M_1$ in $F_1 := \{uv : u \in V \setminus V(M_0), v \in V(M_0)\}$.
After the second pass:
    - $M_1' \leftarrow$ arbitrary largest subset of $M_1$ such that there is no 3-augmenting path in $M_1' \cup M_0$ with respect to $M_0$
    - $V_2 \leftarrow \{x \in V(M_0) : \exists v, w$ such that $vw \in M_1'$ and $wx \in M_0\}$
    - For $x \in V_2$, denote by $P(x)$ the vertex $v$ such that there exists $w$ with $vw \in M_1'$ and $wx \in M_0$. See $x$ and $P(x)$ in Figure 6.
In the third pass: $F_2 := \{xy : x \in V_2, y \in V \setminus V(M_0)\}$
$M_2 \leftarrow \emptyset$
for edge $xy \in F_2$ do
    if $x$, and $y$ are unmarked then
        $M_2 \leftarrow M_2 \cup \{xy\}$; since the graph is triangle free, $y \neq P(x)$, and we can augment $M_0$ using $xy$.
        Mark $P(x)$, $x$, $y$, and $P^{-1}(y)$ (if exists).
Let $M$ be largest of $M_3$ and $M_3'$ which are computed below.
    - Augment $M_0$ using edges in $M_1$ to get $M_3$.
    - Augment $M_0$ using edges in $M_1'$ and $M_2$ to get $M_3'$.
Output $M$.
```

A Three Pass Algorithm for Triangle Free Graphs

For completeness, we present our three-pass algorithm for triangle-free graphs.

▶ **Theorem 14.** *Algorithm 5 is a three-pass, semi-streaming, $(1/2 + 1/10)$-approximation algorithm for maximum matching in triangle-free graphs, and the analysis is tight.*

Proof. Let $|M_0| = (1/2 + \alpha)|M^*|$. The number of edges in M^* incident on $V(M^*) \setminus V(M_0)$ is

$$|V(M^*) \setminus V(M_0)| \geqslant |V(M^*)| - |V(M_0)| = 2|M^*| - 2|M_0| = (1 - 2\alpha)|M^*| ; \tag{7}$$

and these edges also belong to F_1. Since M_1 is a maximal matching in F_1,

$$|M_1| \geqslant (1 - 2\alpha)|M^*|/2 = (1/2 - \alpha)|M^*| . \tag{8}$$

Let c be the number of 3-augmenting paths in $M_1 \cup M_0$, so $|M_1'| = |M_1| - c$ by the definition of M_1'. By Lemma 1, there are at most $4\alpha|M^*|$ non-3-augmentable edges in M_0. So at least $|M_1| - c - 4\alpha|M^*|$ edges of M_1' are incident on 3-augmentable edges of M_0. Therefore there is a matching of size at least $|M_1| - c - 4\alpha|M^*|$ in F_2; consider one such matching M_F. We claim that $|M_2| \geqslant |M_F|/4$. See Figure 6. Let $xy \in M_2$; we note that xy disallows at most four edges in M_F from being added to M_2 due to the (at most) four marks that it adds, because a marked vertex can disallow at most one edge in M_F (due to it being a matching), which shows the claim. Hence:

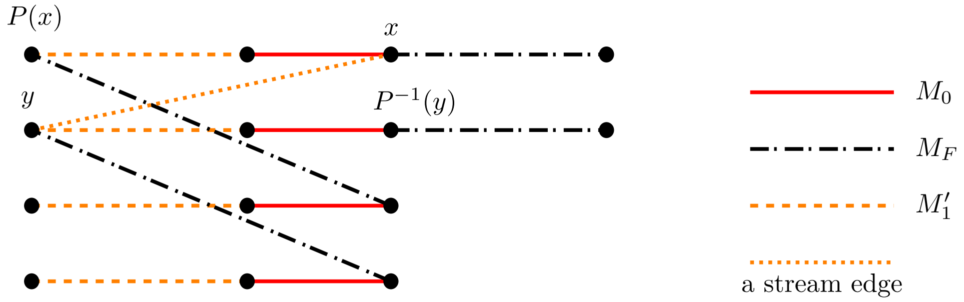


Figure 6 An edge $xy \in M_2$ disallows at most four edges in M_F from being added to M_2.

$$\begin{aligned} |M_2| &\geqslant \frac{|M_F|}{4} \\ &\geqslant \frac{|M_1| - c - 4\alpha|M^*|}{4} \\ &\geqslant \frac{1}{4}\left(\left(\frac{1}{2} - \alpha\right)|M^*| - c - 4\alpha|M^*|\right) && \text{by (8)}, \\ &= \frac{1}{4}\left(\left(\frac{1}{2} - 5\alpha\right)|M^*| - c\right). \end{aligned}$$

Now, each edge in M_2 gives one augmentation after the second pass. To see this, we observe that for any $x \in V_2$, at any point in the algorithm, x and $P(x)$ are either both marked or both unmarked. So when an edge $xy \in M_2$ arrives, x and y are unmarked, and $P(x)$ and $P^{-1}(y)$ (if it exists) are also unmarked, otherwise one of x and y would have been marked and xy would not have been added to M_2. Since both $P(x)$ and $P^{-1}(y)$ were unmarked, we can use the augmenting path $\{M'_1(\{P(x)\}), M_0(\{x\}), xy\}$. Hence we get at least

$$\max\left\{c, \frac{1}{4}\left(\left(\frac{1}{2} - 5\alpha\right)|M^*| - c\right)\right\}$$

augmentations after the third pass. This is minimized by setting

$$\begin{aligned} c &= \frac{1}{4}\left(\left(\frac{1}{2} - 5\alpha\right)|M^*| - c\right) \\ &= \frac{1}{5}\left(\left(\frac{1}{2} - 5\alpha\right)|M^*|\right) \\ &= \left(\frac{1}{10} - \alpha\right)|M^*|. \end{aligned}$$

So we get the following bound:

$$|M| \geqslant |M_0| + \left(\frac{1}{10} - \alpha\right)|M^*| \geqslant \left(\frac{1}{2} + \alpha\right)|M^*| + \left(\frac{1}{10} - \alpha\right)|M^*| = \left(\frac{1}{2} + \frac{1}{10}\right)|M^*|. \blacktriangleleft$$

The tight example is shown in Figure 7.

B Three Pass Algorithm for General Graphs

We find a maximal matching M_1 in the first pass. Then we use Improve-Matching() function from Algorithm 3, i.e.,

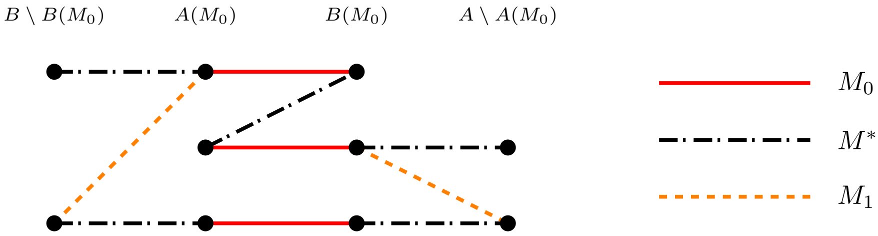


Figure 7 Tight example for Algorithm 5: M_1 has only two edges that land on bad vertices and cannot be augmented in the third pass. So $|M| = |M_0| = 3$ and $|M^*| = 5$.

- in the second pass, $M_2 \leftarrow$ Improve-matching$(M_1, 4, 2)$, and
- in the third pass, $M_3 \leftarrow$ Improve-matching$(M_2, 5, 2)$.

We observe that M_1 is (1/2)-approximate. Then by double application of Lemma 11, we get that M_3 is $(1/2 + 81/1600) \approx (1/2 + 1/19.753)$-approximate.

C Three Pass Algorithm for Bipartite Graphs: Suboptimal Analysis

We now give an analysis of Algorithm 1 that shows approximation ratio of only $1/2 + 1/18$ that is based on Konrad et al.'s [20] analysis for their two-pass algorithm for bipartite graphs. Afterward, we demonstrate that by not considering the distribution of lengths of augmenting paths, we may prove an approximation ratio of at most $1/2 + 1/14$. The better and tight analysis appears in Section 3.

▶ **Theorem 15.** *Algorithm 1 is a three-pass, semi-streaming,* $(1/2 + 1/18)$*-approximation algorithm for maximum matching in bipartite graphs.*

Proof. As usual, let $|M_0| = (1/2 + \alpha)|M^*|$. Since M_0 is a maximal matching, there are $|B(M^*) \setminus B(M_0)|$ edges of M^* that are also in F_2. We have

$$|B(M^*) \setminus B(M_0)| \geqslant |B(M^*)| - |B(M_0)| = |M^*| - |M_0|\,,$$

and since M_A is maximal, we then get the following:

$$|M_A| \geqslant \frac{1}{2}|B(M^*) \setminus B(M_0)| \geqslant \frac{1}{2}(|M^*| - |M_0|) = \frac{1}{2}\left(1 - \left(\frac{1}{2} + \alpha\right)\right)|M^*| = \frac{1}{2}\left(\frac{1}{2} - \alpha\right)|M^*|. \tag{9}$$

By Lemma 1, there are at most $4\alpha|M^*|$ non-3-augmentable edges in M_0. Which means that at least $|M_A| - 4\alpha|M^*|$ edges of M_A are incident on 3-augmentable edges of M_0; therefore there is a matching of size at least $|M_A| - 4\alpha|M^*|$ in F_3. Since we output a maximal matching in F_3, we get at least $(1/2)(|M_A| - 4\alpha|M^*|)$ augmentations after the third pass. So we get the following bound:

$$\begin{aligned}
|M| &\geqslant |M_0| + \frac{1}{2}(|M_A| - 4\alpha|M^*|) \\
&\geqslant |M_0| + \frac{1}{2}\left(\frac{1}{2}\left(\frac{1}{2} - \alpha\right) - 4\alpha\right)|M^*| && \text{by (9)}, \\
&= |M_0| + \left(\frac{1}{8} - \frac{9}{4}\alpha\right)|M^*| \\
&= \left(\frac{1}{2} + \alpha\right)|M^*| + \left(\frac{1}{8} - \frac{9}{4}\alpha\right)|M^*| && \text{because } |M_0| = (1/2+\alpha)|M^*|, \\
&= \left(\frac{1}{2} + \frac{1}{8} - \frac{5}{4}\alpha\right)|M^*|.
\end{aligned}$$

We also have $|M| \geqslant |M_0| = (1/2 + \alpha)|M^*|$. As α increases, the former bound deteriorates and the latter improves, so the worst case α is when these two bounds are equal, which happens at $\alpha = 1/18$, and the approximation ratio we get is $1/2 + 1/18$. ◀

C.1 Improved Analysis Without Considering Longer Augmenting Paths

We can analyze Algorithm 1 better if we bound $|M_A|$ more carefully. The claim is that at least $(1/2 - 7\alpha)|M^*|/2$ edges of M_A are incident on 3-augmentable edges of M_0. Let $A_G \subseteq A(M_0)$ be the set of vertices in A that are endpoints of 3-augmentable edges of M_0; also, let $A_N = A(M_0) \setminus A_G$. So there is a matching of size at least $|A_G|$ in F_2 that covers A_G. Any maximal matching in F_2 has at least $(|A_G| - |A_N|)/2$ edges that are incident on A_G. To see the claim, we use the facts $|A_G| \geqslant (1/2 - 3\alpha)|M^*|$ and $|A_N| \leqslant 4\alpha|M^*|$. So there is a matching of size at least $(1/2 - 7\alpha)|M^*|/2$ in F_3. We output a maximal matching in F_3; hence we get at least $(1/2 - 7\alpha)|M^*|/4$ augmentations after the third pass. So we get the following bound:

$$\begin{aligned}
|M| &\geqslant |M_0| + \frac{1}{4}\left(\frac{1}{2} - 7\alpha\right)|M^*| \\
&= \left(\frac{1}{2} + \alpha\right)|M^*| + \frac{1}{4}\left(\frac{1}{2} - 7\alpha\right)|M^*| \\
&= \left(\frac{1}{2} + \frac{1}{8} - \frac{3}{4}\alpha\right)|M^*|.
\end{aligned}$$

where the second inequality is by (9). We also have $|M| \geqslant |M_0| = (1/2 + \alpha)|M^*|$, so the worst case α is when these two bounds are equal, which happens at $\alpha = 1/14$ and the approximation ratio we get is $1/2 + 1/14$, and we get the following theorem.

▶ **Theorem 16.** *Algorithm 1 is a three-pass, semi-streaming,* $(1/2 + 1/14)$*-approximation algorithm for maximum matching in bipartite graphs.*

D A Note on the Analysis by Esfandiari et al.

We demonstrate with an example that the analysis of the algorithm by Esfandiari et al. [11] given for bipartite graphs cannot be extended for triangle-free graphs to get the same approximation ratio. See Figure 8. Lemma 6 in their paper, as they correctly claim, holds only for bipartite graphs and not for triangle-free graphs. Our algorithm in Section 4 is essentially the same algorithm except for the post-processing step; we augment the maximal matching computed in the first pass greedily, whereas they use an offline maximum matching algorithm. We have highlighted some other comparison points in Section 1.

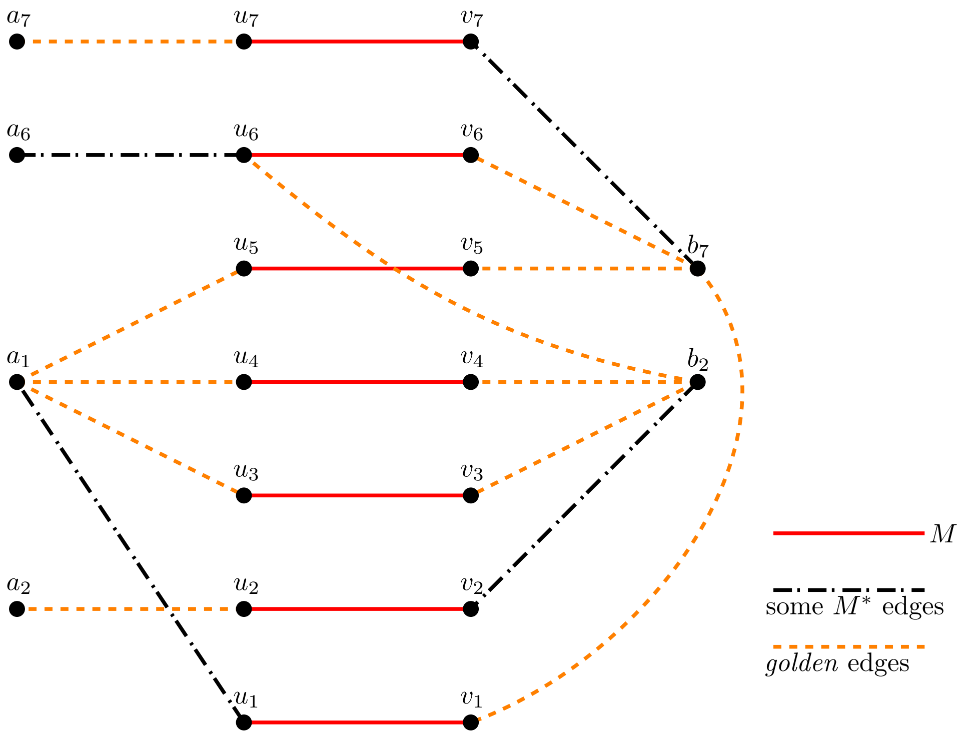


Figure 8 Example demonstrating that Lemma 6 in Esfandiari et al. [11] does not hold when the input graph is not bipartite but is triangle-free. We use $k = 3$. For an M edge u_iv_i, there are two M^* edges incident on it, which are a_iu_i and v_ib_i, and some of the M^* edges are not shown, but all golden edges are shown, which we call support edges or denote by S in our terminology. It can be seen from this example that their algorithm is not a $(1/2 + 1/12)$-approximation algorithm for triangle free graphs, because out of the seven 3-augmentable edges in M, only one will get augmented, thereby giving a worse approximation ratio.

Submodular Secretary Problems: Cardinality, Matching, and Linear Constraints*

Thomas Kesselheim[1] and Andreas Tönnis[2]

1 **Department of Computer Science, TU Dortmund, Dortmund, Germany**[†]
thomas.kesselheim@cs.tu-dortmund.de

2 **Department of Computer Science, University of Bonn, Bonn, Germany**[‡]
atoennis@uni-bonn.de

Abstract

We study various generalizations of the secretary problem with submodular objective functions. Generally, a set of requests is revealed step-by-step to an algorithm in random order. For each request, one option has to be selected so as to maximize a monotone submodular function while ensuring feasibility. For our results, we assume that we are given an offline algorithm computing an α-approximation for the respective problem. This way, we separate computational limitations from the ones due to the online nature. When only focusing on the online aspect, we can assume $\alpha = 1$.

In the *submodular secretary problem*, feasibility constraints are cardinality constraints, or equivalently, sets are feasible if and only if they are independent sets of a k-uniform matroid. That is, out of a randomly ordered stream of entities, one has to select a subset of size k. For this problem, we present a 0.31α-competitive algorithm for all k, which asymptotically reaches competitive ratio α/e for large k. In *submodular secretary matching*, one side of a bipartite graph is revealed online. Upon arrival, each node has to be matched permanently to an offline node or discarded irrevocably. We give a 0.207α-competitive algorithm. This also covers the problem, in which sets of entities are feasible if and only if they are independent with respect to a transversal matroid. In both cases, we improve over previously best known competitive ratios, using a generalization of the algorithm for the classic secretary problem.

Furthermore, we give an $O(\alpha d^{-\frac{2}{B-1}})$-competitive algorithm for submodular function maximization subject to linear packing constraints. Here, d is the column sparsity, that is the maximal number of none-zero entries in a column of the constraint matrix, and B is the minimal capacity of the constraints. Notably, this bound is independent of the total number of constraints. We improve the algorithm to be $O(\alpha d^{-\frac{1}{B-1}})$-competitive if both d and B are known to the algorithm beforehand.

1998 ACM Subject Classification F.1.2 [Models of Computation] Online Computation

Keywords and phrases Secretary Problem, Online Algorithms, Submodular Maximization

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.16

[*] The full version of this article can be found at http://arxiv.org/abs/1607.08805.

[†] Work was done while this author was at Max Planck Institute for Informatics and Saarland University, supported in part by the DFG through Cluster of Excellence MMCI.

[‡] Work was done while this author was at RWTH Aachen University, supported by the DFG GRK/1298 "AlgoSyn".

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 16; pp. 16:1–16:22
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

In the classic secretary problem, one is presented a sequence of items with different scores online in random order. Upon arrival of an item, one has to decide immediately and irrevocably whether to accept or to reject the current item. The objective is to accept the best of these items. Recently, combinatorial generalizations of this problem have attracted attention. In these settings, feasibility of solutions are stated in terms of matroid or linear constraints. In most cases, these combinatorial generalizations consider linear objective functions. This way, the profit gained by the decision in one step is independent of the other steps.

In this paper, we consider general monotone submodular functions[1]. For example, the *submodular secretary problem*, independently introduced by Bateni et al. [4] and Gupta et al. [15], is an online variant of monotone submodular maximization subject to cardinality constraints. In this problem, we are allowed to select up to k items from a set of n items. The value of a set is represented by a monotone, submodular function. Now, stated as an online problem, items arrive one after the other and every item can only be selected right at the moment when it arrives. The values of the submodular function are only known on subsets of the items that have already arrived. The objective function is designed by an adversary, but the order of the items is uniformly at random.

We call an algorithm (asymptotically) c-competitive if for any objective function v chosen by the adversary, the set of selected items ALG satisfies $\mathbf{E}\left[v(\mathrm{ALG})\right] \geq (c - o(1)) \cdot v(\mathrm{OPT})$, where OPT is a size-$k$ subset of items that maximizes v and the $o(1)$-term is asymptotical with respect to the length of the sequence n. Note that any algorithm can pretend n to be larger by adding dummy elements at random positions. Therefore, it is safe to assume that n is large compared to k.

Previous algorithms for submodular secretary problems were designed by modifying offline approximation algorithms for submodular objectives so that they could be used in the online environment [4, 11, 26]. In this paper, we take a different approach. Our algorithms are inspired by algorithms for linear objective functions [17, 18]. We repeatedly solve the respective offline optimization problem and use this outcome as a guide to make decisions in the current round. Generally, it is enough to only compute approximate solutions. Our results nicely separate the loss due to the online nature and due to limited computational power. Using polynomial-time computations and existing offline algorithms, we significantly outperform existing online algorithms. Certain submodular functions or kinds of constraints allow better approximations, which immediately transfer to even better competitive ratios. This is, for example, true for submodular maximization subject to a cardinality constraint if the number of allowed items is constant. Also, if computational complexity is no concern like in classical competitive analysis, our competitive ratios become even better.

1.1 Our Contribution

Given an α-approximate algorithm for monotone submodular maximization subject to a cardinality constraint, we present an $\frac{\alpha}{e}\left(1 - \frac{\sqrt{k-1}}{(k+1)\sqrt{2\pi}}\right)$-competitive algorithm for the submodular secretary problem. That is, we achieve a competitive ratio of at least 0.31α for any $k \geq 2$. Asymptotically for large k, we reach $\frac{\alpha}{e}$.

[1] A function $f\colon 2^U \to \mathbb{R}$ for given ground set U is called *submodular* if for all $S \subseteq T \subseteq U$ and every $x \in U \backslash T$ holds $f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$. Additionally for all sets $S, T \subseteq U$, we call $f(S|T) = f(S \cup T) - f(T)$ the marginal gain of S to T.

Our algorithm follows the following natural paradigm. We reject the first $\frac{n}{e}$ items. Afterwards, for each arriving item, we solve the offline optimization problem of the instance that we have seen so far. If the current item is included in this solution and we have not yet accepted too many items, we accept it. Otherwise, we reject it. For the analysis, we bound the expected value obtained by the algorithm recursively. It then remains to solve the recursion and to bound the resulting term. Generally, the recursive approach can be used for any secretary problems with cardinality constraints. It could be of independent interest, especially because it allows to obtain very good bounds also for rather small values of k.

One option for the black-box offline algorithm is the standard greedy algorithm by Nemhauser and Wolsey [29]. It always picks the item of maximum marginal increase until it has picked k items. Generally, this algorithm is $1-\frac{1}{e}$-approximate. However, it is known that if one compares to the best solution with only $k' \leq k$ items the approximation factor improves to $1 - \exp\left(-\frac{k}{k'}\right)$. We exploit this fact to give a better analysis of our online algorithm when using the greedy algorithm in each step. We show that the algorithm is 0.238-competitive for any k and asymptotically for large k it is 0.275-competitive.

Additionally, we consider the *submodular secretary matching problem*. In this problem, one side of a bipartite graph arrives online in random order. Upon arrival, vertices are either matched to a free vertex on the offline side or rejected. The objective is a submodular function on the set of matched pairs or edges. It is easy to see that the submodular secretary problem is a special case of this more general problem. Fortunately, similar algorithmic ideas work here as well. Again, we combine a sampling phase with a black box for the offline problem and get an 0.207α-competitive algorithm. Notably, the analysis turns out to be much simpler compared to the submodular secretary algorithm.

Finally, we show how our new analysis technique can be used to generalize previous results on linear packing programs towards submodular maximization with packing constraints. Here, we use a typical continuous extension towards the expectation on the submodular objective. We parameterize our results in d, the column sparsity of the constraint matrix, and B, the minimal capacity of the constraints. We achieve a competitive ratio of $\Omega(\alpha d^{-\frac{2}{B-1}})$ if both parameters are not known to the algorithm. If d and B are known beforehand we give different algorithm that is $\Omega(\alpha d^{-\frac{1}{B-1}})$-competitive.

1.2 Related Work

Although the secretary problem itself dates back to the 1960s, combinatorial generalizations only gained considerable interest within the last 10 years. One of the earliest combinatorial generalizations and probably the most famous one is the *matroid secretary problem*, introduced by Babaioff et al. [3]. Here, one has to pick a set of items from a randomly ordered sequence that is an independent set of a matroid. The objective is to maximize the sum of weights of all items picked. It is still believed that there is an $\Omega(1)$-competitive algorithm for this problem; the currently best known algorithms achieve a competitive ratio of $\Omega(1/\log\log(\rho))$ for matroids of rank ρ [13, 24]. Additionally, there are constant competitive algorithms known for many special cases, e.g., for transversal matroids there is an $1/e$-competitive algorithm [17] and for k-uniform matroids there is an $1 - O(1/\sqrt{k})$-competitive algorithm [19]. Both are known to be optimal. Other examples include graphical matroids, for which there is a $1/2e$-competitive algorithm [21], and laminar matroids, for which a $1/9.6$-competitive algorithm is known [26]. Further well-studied generalizations feature *linear constraints*. This includes online packing LPs [8, 27, 2, 18] and online edge-weighted matching [17, 21], for which optimal algorithms are known. Also the online variant of the generalized assignment problem [18] has been studied.

All these secretary problems have in common that the objective function is linear. Compared to other objective functions this has the clear advantage that the gain due to a choice in one round is independent of choices in other rounds. Interdependencies between the rounds only arise due to the constraints. Bateni et al. [4] and Gupta et al. [15] independently started work on submodular objective functions in the secretary setting. To this point, the best known results are a $\frac{e-1}{e^2+e} \approx 0.170$-competitive algorithm for k-uniform matroids [11] and a $\frac{1}{95}$-competitive algorithm for submodular secretary matching [26]. In case there are m linear packing constraints, the best known algorithm is $O(\frac{1}{m})$-competitive [4]. For matroid constraints, Feldman and Zenklusen [14] give a reduction, turning a c-competitive algorithm for linear objective functions to an $\Omega(c^2)$-competitive one for submodular objective functions. Furthermore, they give the first $\Omega(1/\log\log\rho)$-competitive algorithm for the submodular matroid secretary problem. Feldman and Izsak [10] consider more general objective functions, which are not necessarily submodular. They give competitive algorithms for cardinality constraint secretary problems that are parameterized in the *supermodular degree* of the objective function.

Agrawal and Devanur [1] study concave constraints and concave objective functions. These results, however, do not generalize submodular objectives because they require the dimension of the vector space to be low. Representing an arbitrary submodular function would require the dimension to be as large as n. Another related problem is submodular welfare maximization. In this case, even the greedy algorithm is known to be $1/2$-competitive in adversarial order, which is optimal [16], but at least 0.505-competitive in random order [20].

In the offline setting, submodular function maximization is computationally hard if the function is given through a value oracle. There are efficient algorithms that approximate a monotone, submodular function over a matroid or under a knapsack-constraint with a factor of $(1 - 1/e)$ [7, 30]. As a special case, the generalized assignment problem can also be efficiently approximated up to a factor of $(1 - 1/e)$ [7]. For a constant number of linear constraints, there is also a $(1-\epsilon)(1-1/e)$-approximation algorithm [23]. In the non-monotone domain, a number of recent results achieve approximation guarantees close to but strictly better than $1/e$ [6, 9, 5].

2 Submodular Secretary Problem

Let us first turn to the submodular secretary problem. Here, a set of items from a universe U, $|U| = n$, is presented to the algorithm in random order. For each arriving $j \in U$, the algorithm has to decide whether to accept or to reject it, being allowed to accept up to k items in total. The objective is to maximize a monotone submodular function $v\colon 2^U \to \mathbb{R}_{\geq 0}$. This function is defined by an adversary and known to the algorithm only restricted to the subsets of items that have already arrived. This problem extends the secretary problem for k-uniform matroids with linear objective functions, which was solved by Kleinberg [19]. The previously best known competitive factor is $\frac{e-1}{e^2+e} \approx 0.170$ [11].

Depending on the kind of the submodular function and its representation, the corresponding offline optimization problem (monotone submodular maximization with cardinality constraint) can be computationally hard. In order to focus on the online nature of the problem, we assume that we are given an offline algorithm $\mathcal{A}$ that for any $L \subseteq U$ returns an α-approximation of the best solution within L. Formally, $v(\mathcal{A}(L)) \geq \alpha \max_{T\subseteq L, |T|\leq k} v(T)$. Note that $\mathcal{A}$ is allowed to exploit any additional structure of the function v. For different L and L', $\mathcal{A}(L)$ and $\mathcal{A}(L')$ do not have to be consistent, but the output $\mathcal{A}(L)$ must be identical, irrespective of the arrival order on L. It may also be randomized. In this case, let $v(\mathcal{A}(L))$ refer to the *expected* value achieved on set L.

Algorithm 1: Submodular k-secretary

```
Drop the first $\lceil pn \rceil - 1$ items;
for item $j$ arriving in round $\ell \geq \lceil pn \rceil$ do        // online steps $\ell = \lceil pn \rceil$ to $n$
    Set $U^{\leq \ell} := U^{\leq \ell - 1} \cup \{j\}$;
    Let $S^{(\ell)} = \mathcal{A}(U^{\leq \ell})$;                     // black box $\alpha$-approximation
    if $j \in S^{(\ell)}$ then                                          // tentative allocation
        if $|\text{Accepted}| < k$ then                                 // feasibility test
            Add $j$ to Accepted;                                        // online allocation
```

Our online algorithm, Algorithm 1, uses algorithm $\mathcal{A}$ as a subroutine as follows. It starts by rejecting the first pn items. For every following item j, it runs $\mathcal{A}(L)$, where L is the set of items that have arrived up to this point. If $j \in \mathcal{A}(L)$ we call j tentatively selected. Furthermore if the set of accepted items S contains less than k items and j is tentatively selected, then the algorithm adds j to S. Otherwise, it rejects j.

▶ **Theorem 1.** *Algorithm 1 for the submodular secretary problem is* $\frac{\alpha}{e}\left(1 - \frac{\sqrt{k-1}}{(k+1)\sqrt{2\pi}}\right)$*-competitive with sample size* $pn = \frac{n}{e}$.

2.1 Analysis Technique

Before proving Theorem 1, let us shed some light on the way we lower-bound the value of the submodular objective function. To this end, we consider the expected value of the set of all tentatively selected items T. In other words, we pretend all selections our algorithm tries to make are actually feasible. It seems natural to bound the expected value of T by adding up the marginal gains round-by-round given the tentative selections in earlier rounds. Unfortunately, this introduces complicated dependencies on the order of arrival of previous items. Therefore, we take a different approach and bound the respective marginal gains with respect of tentative selections in *future* rounds. The important insight is that this keeps the dependencies manageable.

▶ **Proposition 2.** *The set of all items* T *that are tentatively selected by Algorithm 1 has an expected value of* $\mathbf{E}[v(T)] \geq \left(\frac{\alpha}{e} - \frac{\alpha}{n}\right) \cdot v(\mathrm{OPT})$ *if the algorithm is run with sample size* $pn = \frac{n}{e}$.

Proof. Let $T^{\geq \ell}$ denote the set of tentatively selected items that arrive in or after round ℓ. Formally, we have $T^{\geq \ell} = \{j\} \cup T^{\geq \ell+1}$ if $j \in \mathcal{A}(U^{\leq \ell})$ and $T^{\geq \ell} = T^{\geq \ell+1}$ otherwise.

We consider a different random process to define the $T^{\geq \ell}$ random variables, which results in the same distribution. First, we draw one item from U uniformly to come last. This determines the value of $T^{\geq n}$. Then we continue by drawing on item out of the remaining ones to come second to last, determining $T^{\geq n-1}$. Generally, this means that conditioning on $U^{\leq \ell}$ and the values of $T^{\geq \ell'}$, for $\ell' > \ell$, the item j is drawn uniformly at random from $U^{\leq \ell}$ and the respective outcome determines $T^{\geq \ell}$.

We bound the expected tentative value collected in rounds ℓ to n conditioned on the items that arrived before round ℓ and conditioned on all items that are tentatively selected. Through this condition, the value of the sets $T^{\geq \ell+1}$ to $T^{\geq n}$ is already fixed. The expectation is only over the marginal gain of j with respect to the future tentatively selected items $T^{\geq \ell+1}$

$$\mathbf{E}\left[v(T^{\geq \ell}) \;\middle|\; U^{\leq \ell}, T^{\geq \ell'} \text{for all } \ell' > \ell\right] = \frac{1}{\ell}\Big(\sum_{j \in \mathcal{A}(U^{\leq \ell})} v(\{j\}|T^{\geq \ell+1})\Big) + v(T^{\geq \ell+1}) \enspace .$$

Due to submodularity, the gain of the set $\mathcal{A}(U^{\leq \ell})$ is at most the sum of the individual marginal gains of the items in $\mathcal{A}(U^{\leq \ell})$. This gives us

$$\sum_{j \in \mathcal{A}(U^{\leq \ell})} v(\{j\}|T^{\geq \ell+1}) \geq v\left(\mathcal{A}(U^{\leq \ell}) \mid T^{\geq \ell+1}\right) \geq v\left(\mathcal{A}(U^{\leq \ell})\right) - v(T^{\geq \ell+1}) \enspace .$$

In the last inequality, we use monotony of the objective function. This yields

$$\mathbf{E}\left[v(T^{\geq \ell}) \;\middle|\; U^{\leq \ell}, T^{\geq \ell'} \text{for all } \ell' > \ell\right] \geq \frac{1}{\ell} v\left(\mathcal{A}(U^{\leq \ell})\right) + \left(1 - \frac{1}{\ell}\right) v(T^{\geq \ell+1}) \enspace .$$

We take the expectation over the remaining randomization and get the following recursion

$$\mathbf{E}\left[v(T^{\geq \ell})\right] \geq \frac{1}{\ell}\mathbf{E}\left[v\left(\mathcal{A}(U^{\leq \ell})\right)\right] + \left(1 - \frac{1}{\ell}\right)\mathbf{E}\left[v(T^{\geq \ell+1})\right] \enspace .$$

Observe that $\mathrm{OPT} \cap U^{\leq \ell}$ is fully contained in $U^{\leq \ell}$ and has size at most k. Therefore, the approximation guarantee of $\mathcal{A}$ yields that $v(\mathcal{A}(U^{\leq \ell})) \geq \alpha v(\mathrm{OPT} \cap U^{\leq \ell})$. Furthermore, submodularity gives us $\mathbf{E}\left[v(\mathrm{OPT} \cap U^{\leq \ell})\right] \geq \frac{\ell}{n} v(\mathrm{OPT})$ because each item is included in $U^{\leq \ell}$ with probability $\frac{\ell}{n}$. In combination, this gives us

$$\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] \geq \alpha \mathbf{E}\left[v(\mathrm{OPT} \cap U^{\leq \ell})\right] \geq \alpha \frac{\ell}{n} v(\mathrm{OPT}) \enspace . \tag{1}$$

Now we solve the recursion

$$\mathbf{E}\left[v(T^{\geq \ell})\right] \geq \frac{\alpha}{n} v(\mathrm{OPT}) + \left(1 - \frac{1}{\ell}\right)\mathbf{E}\left[v(T^{\geq \ell+1})\right] = \sum_{j=\ell}^{n} \prod_{i=\ell}^{j-1} \left(1 - \frac{1}{i}\right) \frac{\alpha}{n} v(\mathrm{OPT}) \enspace .$$

We have $\prod_{i=\ell}^{j-1}\left(1 - \frac{1}{i}\right) = \frac{\ell-1}{j-1}$ and $\sum_{j=\ell}^{n} \frac{1}{j-1} \geq \ln(\frac{n}{\ell})$ for all $\ell \geq 2$. This yields

$$\mathbf{E}\left[v(T^{\geq \ell})\right] \geq \sum_{j=\ell}^{n} \prod_{i=\ell}^{j-1} \left(1 - \frac{1}{i}\right) \frac{\alpha}{n} v(\mathrm{OPT}) = \frac{\alpha}{n} v(\mathrm{OPT}) \sum_{j=\ell}^{n} \frac{\ell - 1}{j - 1} \geq \frac{\ell - 1}{n} \ln\left(\frac{n}{\ell}\right) \alpha v(\mathrm{OPT}) \enspace .$$

With $\ell = pn$ and sample size $pn = \frac{n}{e}$, we get

$$\mathbf{E}\left[v(T^{\geq pn})\right] \geq \frac{pn - 1}{n} \ln\left(\frac{1}{p}\right) \alpha v(\mathrm{OPT}) = \left(\frac{1}{e} - \frac{1}{n}\right) \alpha v(\mathrm{OPT}) \enspace . \qquad ◀$$

The probability of a tentative selection in round ℓ is $\frac{k}{\ell}$. This means, in expectation, we make $\sum_{\ell=\frac{n}{e}}^{n} \frac{k}{\ell} \approx k$ tentative selections. Therefore, for large values of k, it is likely that most tentative selections are feasible. This way, we could already derive guarantees for large k. However, for small k, the derived bound would be far to pessimistic. This is due to the fact that we bound the marginal gain of an item based on all *tentative* future ones. If some of them are indeed not feasible, we underestimate the contribution of earlier items. Therefore, Theorem 1 requires a more involved recursion that is based on the idea from this section, but also incorporates the probability that an item is feasible directly.

2.2 Proof of Theorem 1

To prove the theorem, we will derive a lower bound on the value collected by the algorithm starting from an arbitrary round $\ell \in [n]$ with an arbitrary remaining capacity $r \in \{0, 1, \ldots, k\}$. The random variables $\mathrm{ALG}_r^{\geq \ell} \subseteq U$ represent the set of first r items that a hypothetical run of the algorithm would collect if it started the *for* loop of Algorithm 1 in round ℓ. Formally, we define them recursively as follows. We set $\mathrm{ALG}_0^{\geq \ell} = \emptyset$ for all ℓ and $\mathrm{ALG}_r^{\geq n+1} = \emptyset$ for all r. For $\ell \in [n]$, $r > 0$, let j be the item arriving in round ℓ, and $U^{\leq \ell}$ be the set of items arriving until and including round ℓ. We define $\mathrm{ALG}_r^{\geq \ell} = \{j\} \cup \mathrm{ALG}_{r-1}^{\geq \ell+1}$ if $j \in \mathcal{A}(U^{\leq \ell})$ and $\mathrm{ALG}_r^{\geq \ell} = \mathrm{ALG}_r^{\geq \ell+1}$ otherwise. Note that by this definition $\mathrm{ALG} = \mathrm{ALG}_k^{\geq pn}$. Furthermore, for every possible arrival order, $\mathrm{ALG}_r^{\geq \ell}$ is pointwise a superset of $\mathrm{ALG}_{r-1}^{\geq \ell}$ for $r > 0$.

In Lemma 3, we show a recursive lower bound on the value of these sets. In this part, the precise definition of $\mathrm{ALG}_r^{\geq \ell}$ will be crucial to avoid complex dependencies. Afterwards, in Lemma 4, we solve this recursion. Given this solution, we can finally prove Theorem 1.

▶ **Lemma 3.** *For all $\ell \in [n]$ and $r \in \{0, 1, \ldots, k\}$, we have*

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq \frac{1}{\ell}\left(\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] + (k-1)\mathbf{E}\left[v(\mathrm{ALG}_{r-1}^{\geq \ell+1})\right] + (\ell - k)\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell+1})\right]\right) .$$

Proof. As explained in Section 2.1, we first draw one item from U uniformly at random to be the item that arrives in round n. This defines the values of $\mathrm{ALG}_r^{\geq n}$ for all r. Then we draw another item to be the second to last one and so on. In this way, we can condition on $U^{\leq \ell}$ and the values of $\mathrm{ALG}_r^{\geq \ell'}$, for $\ell' > \ell$ and all r. In round ℓ, the item j is drawn uniformly at random from $U^{\leq \ell}$ and the respective outcome determines $\mathrm{ALG}_r^{\geq \ell}$ for all r. This allows us to write for $r > 0$

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell}) \;\middle|\; U^{\leq \ell}, \mathrm{ALG}_{r'}^{\geq \ell'} \text{ for all } \ell' > \ell \text{ and all } r'\right]$$

$$v = \frac{1}{\ell}\left(\sum_{j \in \mathcal{A}(U^{\leq \ell})} v(\{j\} \cup \mathrm{ALG}_{r-1}^{\geq \ell+1}) + |U^{\leq \ell} \setminus \mathcal{A}(U^{\leq \ell})| v(\mathrm{ALG}_r^{\geq \ell+1})\right) .$$

By submodularity, we have

$$\sum_{j \in \mathcal{A}(U^{\leq \ell})} \left(v(\{j\} \cup \mathrm{ALG}_{r-1}^{\geq \ell+1}) - v(\mathrm{ALG}_{r-1}^{\geq \ell+1})\right) \geq v(\mathcal{A}(U^{\leq \ell}) \cup \mathrm{ALG}_{r-1}^{\geq \ell+1}) - v(\mathrm{ALG}_{r-1}^{\geq \ell+1}) ,$$

and hence

$$\sum_{j \in \mathcal{A}(U^{\leq \ell})} v(\{j\} \cup \mathrm{ALG}_{r-1}^{\geq \ell+1}) \geq v(\mathcal{A}(U^{\leq \ell}) \cup \mathrm{ALG}_{r-1}^{\geq \ell+1}) + (|\mathcal{A}(U^{\leq \ell})| - 1) v(\mathrm{ALG}_{r-1}^{\geq \ell+1}) .$$

This gives us

$$\begin{aligned} &\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell}) \;\middle|\; U^{\leq \ell}, \mathrm{ALG}_{r'}^{\geq \ell'} \text{ for all } \ell' > \ell \text{ and all } r'\right] \\ &\quad\geq \frac{1}{\ell} v(\mathcal{A}(U^{\leq \ell}) \cup \mathrm{ALG}_{r-1}^{\geq \ell+1}) + \frac{|\mathcal{A}(U^{\leq \ell})| - 1}{\ell} v(\mathrm{ALG}_{r-1}^{\geq \ell+1}) \\ &\qquad + \frac{|U^{\leq \ell} \setminus \mathcal{A}(U^{\leq \ell})|}{\ell} v(\mathrm{ALG}_r^{\geq \ell+1}) . \end{aligned}$$

Furthermore, by applying the monotonicity of v and the facts that $|\mathcal{A}(U^{\leq \ell})| \leq k$ and $\mathrm{ALG}_{r-1}^{\geq \ell+1} \subseteq \mathrm{ALG}_r^{\geq \ell+1}$, we get

$$\begin{aligned} &\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell}) \;\middle|\; U^{\leq \ell}, \mathrm{ALG}_{r'}^{\geq \ell'} \text{ for all } \ell' > \ell \text{ and all } r'\right] \\ &\quad\geq \frac{1}{\ell}\left(v(\mathcal{A}(U^{\leq \ell})) + (k-1) v(\mathrm{ALG}_{r-1}^{\geq \ell+1}) + (\ell - k) v(\mathrm{ALG}_r^{\geq \ell+1})\right) . \end{aligned}$$

Taking the expectation over all remaining randomization yields

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq \frac{1}{\ell}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] + \frac{k-1}{\ell}\mathbf{E}\left[v(\mathrm{ALG}_{r-1}^{\geq \ell+1})\right] + \frac{\ell-k}{\ell}\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell+1})\right] . ◀$$

The next step is to solve the recursion.

▶ **Lemma 4.** *For all $\ell \in [n]$, $\ell \geq k^2 + k$, and $r \in \{0, 1, \dots, k\}$, we have*

$$\frac{\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right]}{v(\mathrm{OPT})} \geq \left(\frac{r\ell}{(k-1)n} - \frac{1}{k-1}\left(\frac{\ell}{n}\right)^k \sum_{r'=0}^{r-1}\sum_{i=0}^{r'} \frac{(k-1)^i}{i!} \ln^i\left(\frac{n}{\ell}\right) - \frac{3k^2 r}{(k-1)n} \right) \alpha \ . \quad (2)$$

Proof (Outline). As a first step, we eliminate the recursive reference from $\mathrm{ALG}_r^{\geq \ell}$ to $\mathrm{ALG}_r^{\geq \ell+1}$. To this end, we count the rounds until the next item is accepted. Repeatedly inserting the bound for $\mathrm{ALG}_r^{\geq \ell+1}$ into the one for $\mathrm{ALG}_r^{\geq \ell}$ gives us

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq \sum_{j=\ell}^{n} \left(\prod_{i=\ell}^{j-1} \left(1 - \frac{k}{i}\right) \left(\frac{k-1}{j}\mathbf{E}\left[v(\mathrm{ALG}_{r-1}^{\geq j+1})\right] + \frac{1}{j}\mathbf{E}\left[v(\mathcal{A}(U^{\leq j}))\right] \right) \right) \ .$$

With Equation (1) in Section 2.1 we have $\mathbf{E}\left[v(\mathcal{A}(U^{\leq j}))\right] \geq \frac{j}{n}\alpha v(\mathrm{OPT})$.

We use $\prod_{i=\ell}^{j-1}\left(1 - \frac{k}{i}\right) = \frac{(\ell-1)!}{(\ell-k-1)!}\frac{(j-k-1)!}{(j-1)!} \geq \left(\frac{\ell-k}{j-k}\right)^k$ and get

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq \sum_{j=\ell}^{n} \left(\left(\frac{\ell-k}{j-k}\right)^k \left(\frac{k-1}{j+1}\mathbf{E}\left[v(\mathrm{ALG}_{r-1}^{\geq j+1})\right] + \frac{\alpha}{n}v(\mathrm{OPT}) \right) \right) \ . \quad (3)$$

It can be shown that (2) provides a lower bound on the functions defined by this recursion. For details, see Appendix A.1. ◀

Proof of Theorem 1. To complete the proof of the theorem, we apply Lemma 4 for $\ell = pn$ and $r = k$. This gives us $\mathbf{E}\left[v(\mathrm{ALG})\right] = \mathbf{E}\left[v(\mathrm{ALG}_k^{\geq pn})\right]$ and thus

$$\mathbf{E}\left[v(\mathrm{ALG})\right] \geq \left(\frac{pk}{k-1} - \frac{1}{k-1}p^k \sum_{r'=0}^{k-1}\sum_{i=0}^{r'} \frac{(k-1)^i}{i!} \ln^i\left(\frac{1}{p}\right) - \frac{6k^2}{n} \right) \cdot \alpha v(\mathrm{OPT}) \ .$$

For p such that $pn = \lceil \frac{n}{e} \rceil$, we have $p \leq \frac{1}{e} + \frac{1}{n}$ and $\ln\left(\frac{1}{p}\right) = 1 + \ln\left(\frac{n}{n+e}\right) \leq 1$. For sake of readability, we omit the error term in the remainder of the proof. The more detailed calculation is included in Appendix A.2. With $p = \frac{1}{e}$, we have $\ln\left(\frac{1}{p}\right) = 1$, this allows us to reorder the double sum as follows

$$\sum_{r'=0}^{k-1}\sum_{i=0}^{r'} \frac{(k-1)^i}{i!} = \sum_{i=0}^{k-1}(k-i)\frac{(k-1)^i}{i!} = \sum_{i=0}^{k-1}\frac{(k-1)^i}{i!} + \frac{(k-1)^k}{(k-1)!} \ .$$

By definition of the exponential function $e^x = \sum_{i=0}^{\infty}\frac{x^i}{i!}$. For $x > 0$, all terms of the infinite sum are positive. This yields $e^x \geq \sum_{i=0}^{k-1}\frac{x^i}{i!} + \frac{x^k}{k!} + \frac{x^{k+1}}{(k+1)!}$ and thus by setting $x = k-1$ we get

$$\sum_{r'=0}^{k-1}\sum_{i=0}^{r'} \frac{(k-1)^i}{i!} \leq e^{k-1} - \frac{(k-1)^k}{k!} - \frac{(k-1)^{k+1}}{(k+1)!} + \frac{(k-1)^k}{(k-1)!} \ .$$

This implies

$$\frac{\mathbf{E}\left[v(\mathrm{ALG})\right]}{\alpha v(\mathrm{OPT})} \geq \frac{k}{e(k-1)} - \frac{1}{e^k(k-1)}\left(e^{k-1} - \frac{(k-1)^k}{k!} - \frac{(k-1)^{k+1}}{(k+1)!} + \frac{(k-1)^k}{(k-1)!}\right) - \frac{6k^2}{n}$$
$$= \frac{1}{e} - \frac{1}{e^k}\frac{k-1}{k+1}\frac{(k-1)^{k-1}}{(k-1)!} - \frac{6k^2}{n} \quad .$$

It only remains to apply the Stirling approximation $(k-1)! \geq \sqrt{2\pi(k-1)}\left(\frac{k-1}{e}\right)^{k-1}$ to get

$$\frac{\mathbf{E}\left[v(\mathrm{ALG})\right]}{\alpha v(\mathrm{OPT})} \geq \frac{1}{e}\left(1 - \frac{\sqrt{k-1}}{(k+1)\sqrt{2\pi}}\right) - \frac{6k^2}{n} \quad . \qquad ◀$$

2.3 Improved Analysis for the Greedy Algorithm

One possible choice for the algorithm $\mathcal{A}$ is the greedy algorithm by Nemhauser and Wolsey [29]. It repeatedly picks the item with the highest marginal increase compared to the items chosen so far until k items have been picked. As pointed out in [22], the approximation guarantee would improve further when picking more items according to the greedy rule. In other words, if we let our algorithm pick k elements but compare the outcome to the optimal solution of only k' items, the approximation factor improves to $1 - \exp\left(-\frac{k}{k'}\right)$.

We can exploit this fact in the analysis of the online algorithm that uses the greedy algorithm as $\mathcal{A}$ in Algorithm 1. The reason is that in early rounds only some items of the optimal solution have arrived. Our algorithm, however, always chooses a set of size k for $S^{(\ell)} = \mathcal{A}(U^{\leq \ell})$. In the generic analysis, we show that $\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] \geq \alpha \frac{\ell}{n} v(\mathrm{OPT})$. In case of $\mathcal{A}$ being the greedy algorithm, we can improve this bound as follows.

▶ **Lemma 5.** *$\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] \geq \alpha_\ell \frac{\ell}{n} v(\mathrm{OPT})$ for $\alpha_\ell = 1 - \frac{\ell}{en} - \frac{1}{ek}$.*

Proof. Consider the offline optimum OPT and $\mathrm{OPT} \cap U^{\leq \ell}$, its restriction to the items that arrive by round ℓ. Let $Z = |\mathrm{OPT} \cap U^{\leq \ell}|$ be the number of OPT items that arrive by round ℓ.

Condition on any value of Z. Observe that by symmetry the probably of every OPT item to have arrived by round ℓ is $\frac{Z}{k}$. Therefore, submodularity implies $\mathbf{E}\left[v(\mathrm{OPT} \cap U^{\leq \ell}) \mid Z\right] \geq \frac{Z}{k} v(\mathrm{OPT})$. If the greedy algorithm picks k elements, it achieves value at least $\left(1 - \exp\left(-\frac{k}{Z}\right)\right) \cdot v(\mathrm{OPT} \cap U^{\leq \ell})$. In combination, this gives us $\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell})) \mid Z\right] \geq \left(1 - \exp\left(-\frac{k}{Z}\right)\right) \frac{Z}{k} v(\mathrm{OPT})$. We now use the fact that $\exp\left(\frac{k}{Z}\right) \geq e\frac{k}{Z}$ because $Z \leq k$. Therefore $\exp\left(-\frac{k}{Z}\right) \leq \frac{Z}{ek}$ and $\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell})) \mid Z\right] \geq \left(1 - \frac{Z}{ek}\right) \frac{Z}{k} v(\mathrm{OPT})$.

It remains to take the expectation over Z. We have $\mathbf{E}\left[Z\right] = \frac{\ell}{n} k$. Letting $Z_j = 1$ if $j \in U^{\leq \ell}$ and 0 otherwise, we have and $\mathbf{E}\left[Z^2\right] = \mathbf{E}\left[\sum_{j \in \mathrm{OPT}} Z_j + \sum_{j \in \mathrm{OPT}} \sum_{j' \in \mathrm{OPT}, j' \neq j} Z_j Z_{j'}\right] = \frac{\ell}{n} k + k(k-1)\frac{\ell}{n}\frac{\ell-1}{n-1} \leq \frac{\ell}{n} k + \left(\frac{\ell}{n} k\right)^2$. This implies

$$\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] \geq \left(\frac{\mathbf{E}\left[Z\right]}{k} - \frac{\mathbf{E}\left[Z^2\right]}{ek^2}\right) v(\mathrm{OPT}) \geq \left(\frac{\ell}{n} - \frac{\ell^2}{en^2} - \frac{\ell}{ekn}\right) v(\mathrm{OPT}) \quad . \qquad ◀$$

Given this lemma, we can follow similar steps as in the proof of Theorem 1 to show an improved guarantee of this particular algorithm. In more detail, we get competitive ratios of at least 0.177 for any $k \geq 2$. Asymptotically for large k we reach 0.275.

▶ **Theorem 6.** *If the greedy algorithm is used as blackbox approximation algorithm $\mathcal{A}$, then Algorithm 1 is $\frac{1 + \frac{1}{2e^3} - \frac{3}{2e} - \frac{e-1}{e^2 k}}{e-1}\left(1 - \frac{\sqrt{k-1}}{(k+1)\sqrt{2\pi}}\right)$-competitive with sample size $pn = \frac{n}{e}$.*

To prove Theorem 6, we combine Lemmas 3 and 5, which give us a recursive formula for $\mathrm{ALG}_r^{\geq \ell}$. We first solve the recursion (Claim 7) and then show that the occurring coefficients are non-increasing (Claim 8). This then allows to apply Chebyshev's sum inequality.

▶ **Claim 7.** *Lemma 3 implies*

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq \sum_{j=\ell}^{n} \frac{a_{\ell,j-1}}{j} \mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] \sum_{r'=0}^{r-1} \sum_{\substack{M \subseteq \{\ell,\dots,j-1\} \\ |M|=r'}} \left(\prod_{i \in M} \frac{k-1}{i} \right)$$

with $a_{\ell,j-1} = \prod_{i=\ell}^{j-1}\left(1 - \frac{k}{i}\right)$.

The proof of this claim is by induction and it is included in Appendix A.3.

▶ **Claim 8.** *Let*

$$t_{\ell,j} = a_{\ell,j-1} \sum_{r'=0}^{r-1} \sum_{\substack{M \subseteq \{\ell,\dots,j-1\} \\ |M|=r'}} \left(\prod_{i \in M} \frac{k-1}{i} \right)$$

with $a_{\ell,j-1} = \prod_{i=\ell}^{j-1}\left(1 - \frac{k}{i}\right)$. *For fixed* ℓ*, the sequence* $t_{\ell,j}$ *is non-increasing in* j.

The proof of this claim is included in Appendix A.4.

Proof of Theorem 6. Now we can proceed to the proof of Theorem 6. So far, we have shown that

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq \sum_{j=\ell}^{n} \frac{t_{\ell,j}}{j} \mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] \quad \text{for} \quad t_{\ell,j} = a_{\ell,j-1} \sum_{r'=0}^{r-1} \sum_{\substack{M \subseteq \{\ell,\dots,j-1\} \\ |M|=r'}} \left(\prod_{i \in M} \frac{k-1}{i} \right)$$

with $a_{\ell,j-1} = \prod_{i=\ell}^{j-1}\left(1 - \frac{k}{i}\right)$. Furthermore, Lemma 5 shows that $\frac{\mathbf{E}[v(\mathcal{A}(U^{\leq \ell}))]}{j} \geq \frac{\alpha_j v(\mathrm{OPT})}{n}$ for $\alpha_\ell = 1 - \frac{\ell}{en} - \frac{1}{ek}$.

As both $t_{\ell,j}$ and α_j are non-increasing in j, we can use Chebyshev's sum inequality to get

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq \sum_{j=\ell}^{n} t_{\ell,j} \frac{\alpha_j v(\mathrm{OPT})}{n} \geq \left(\sum_{j=\ell}^{n} t_{\ell,j} \frac{v(\mathrm{OPT})}{n} \right) \left(\frac{1}{n-\ell} \sum_{j=\ell}^{n} \alpha_j \right) .$$

It now remains to bound these two terms.

First, we show that the sum $\sum_{j=\ell}^{n} t_{\ell,j} \frac{c}{n}$ with $c = v(\mathrm{OPT})$ is lower-bounded by a recursion of the form of Equation (3). Similar calculations to Lemma 4 will then give us the respective bound. Similar to the previous proof, we use $a_{\ell,j-1} = \prod_{i=\ell}^{j-1}\left(1 - \frac{k}{i}\right) \geq \left(\frac{\ell-k}{j-k}\right)^k$ and get

$$\begin{aligned} \sum_{j=\ell}^{n} t_{\ell,j} \frac{v(\mathrm{OPT})}{n} &= \sum_{j=\ell}^{n} a_{\ell,j-1} \sum_{r'=0}^{r-1} \sum_{\substack{M \subseteq \{\ell,\dots,j\} \\ |M|=r'}} \left(\prod_{i \in M} \frac{k-1}{i} \right) \frac{c}{n} \\ &\geq \sum_{j=\ell}^{n} \left(\frac{\ell-k}{j-k} \right)^k \sum_{r'=0}^{r-1} \sum_{\substack{M \subseteq \{\ell,\dots,j\} \\ |M|=r'}} \left(\prod_{i \in M} \frac{k-1}{i+1} \right) \frac{c}{n} . \end{aligned}$$

Let now

$$b_{\ell,r'} = \sum_{j=\ell}^{n} \left(\frac{\ell-k}{j-k}\right)^k \sum_{r'=0}^{r-1} \sum_{\substack{M\subseteq\{\ell,\dots,j\}\\|M|=r'}} \left(\prod_{i\in M} \frac{k-1}{i+1}\right) \frac{c}{n} \ .$$

We combine the two inner sums and then pull out the earliest element $m \in M \subseteq \{\ell,\dots,j\}$ recursively. We move the corresponding factor out of the product and get

$$\begin{aligned} b_{\ell,r'} &= \sum_{j=\ell}^{n} \left(\frac{\ell-k}{j-k}\right)^k \sum_{\substack{M\subseteq\{\ell,\dots,j\}\\|M|\leq r'}} \left(\prod_{i\in M} \frac{k-1}{i+1}\right) \frac{c}{n} \\ &= \sum_{j=\ell}^{n} \left(\frac{\ell-k}{j-k}\right)^k \left(\frac{c}{n} + \sum_{m=\ell}^{j-1} \frac{k-1}{m+1} \sum_{\substack{M\subseteq\{m+1,\dots,j\}\\|M|\leq r'-1}} \left(\prod_{i\in M} \frac{k-1}{i+1}\right) \frac{c}{n}\right) \ . \end{aligned}$$

At this point, we change the order of summation such that we sum over m first. We can keep the constant part in place, since both sums $\sum_{j=\ell}^{n} \left(\frac{\ell-k}{j-k}\right)^k = \sum_{m=\ell}^{n} \left(\frac{\ell-k}{m-k}\right)^k$ amount the same. Now the inner part matches the recursion given above

$$\begin{aligned} b_{\ell,r'} &= \sum_{m=\ell}^{n} \left(\frac{\ell-k}{m-k}\right)^k \left(\frac{c}{n} + \frac{k-1}{m} \sum_{j=m+1}^{n} \left(\frac{m-k}{j-k}\right)^k \sum_{\substack{M\subseteq\{m+1,\dots,j\}\\|M|\leq r'-1}} \left(\prod_{i\in M} \frac{k-1}{i}\right) \frac{c}{n}\right) \\ &= \sum_{m=\ell}^{n} \left(\frac{\ell-k}{m-k}\right)^k \left(\frac{c}{n} + \frac{k-1}{m} b_{m+1,r'-1}\right) \ . \end{aligned}$$

From this point on, we follow the proof of Lemma 4 in Appendix A.1 and get the following lemma.

▶ **Lemma 9.** *Given a recursion of the form*

$$b_{\ell,r} = \sum_{j=\ell}^{n} \left(\left(\frac{\ell-k}{j-k}\right)^k \left(\frac{k-1}{j+1} b_{j+1,r-1} + \frac{c}{n}\right)\right)$$

with $b_{n+1,r} = 0$ *and* $b_{\ell,0} = 0$*. Then*

$$b_{\ell,r} \geq \left(\frac{r(\ell-k)}{(k-1)n} - \frac{1}{k-1} \left(\frac{\ell-k}{n-k}\right)^k \sum_{r'=0}^{r-1} \sum_{i=0}^{r'} \frac{(k-1)^i}{i!} \ln^i\left(\frac{n}{\ell}\right) - \frac{3k^2 r}{(k-1)n} \right) c \ .$$

Consequently, following the calculations in the proof of Theorem 1

$$\mathbf{E}\left[v(\mathrm{ALG})\right] = \mathbf{E}\left[v(\mathrm{ALG}_k^{\geq n/e})\right] \geq \frac{1}{e}\left(1 - \frac{\sqrt{k-1}}{(k+1)\sqrt{2\pi}} - \frac{6ek^2}{n}\right) \left(\frac{1}{n - n/e} \sum_{j=n/e}^{n} \alpha_j\right) v(\mathrm{OPT}).$$

For $\alpha_j = 1 - \frac{j}{en} - \frac{1}{ek}$, we can bound the last term through the integral and get

$$\frac{1}{n - n/e} \sum_{j=n/e}^{n} \left(1 - \frac{j}{en} - \frac{1}{ek}\right) \geq \frac{1}{1 - 1/e} \left(1 + \frac{1}{2e^3} - \frac{3}{2e} - \frac{e-1}{e^2 k}\right) \ .$$

For large k, we have an asymptotic competitive ratio of $\frac{1}{e}\left(1 + \frac{1}{2e^3} - \frac{3}{2e}\right) \approx 0.275$. ◀

Algorithm 2: Submodular Bipartite Online Matching

```
Drop the first $\lceil pn \rceil - 1$ vertices;
for vertex $u \in L$ in round $\ell \geq \lceil pn \rceil$ do          // online steps $\ell = \lceil pn \rceil$ to $n$
    Set $L^{\leq \ell} := L^{\leq \ell - 1} \cup \{u\}$;
    Let $M^{(\ell)} = \mathcal{A}(L^{\leq \ell} \cup R)$;                // black box $\alpha$-approximation
    Let $e^{(\ell)} := (u, r)$ be the edge assigned to $u$ in $M^{(\ell)}$;     // tentative edge
    if Accepted $\cup\, e^{(\ell)}$ is a matching then                  // feasibility test
        Add $e^{(\ell)}$ to Accepted;                                 // online allocation
```

3 Submodular Matching

Next, we consider the online submodular bipartite matching problem. In the offline version, we are given a bipartite graph $G = (L \cup R, E)$ and a monotone, submodular, non-decreasing objective function $v \colon 2^E \to \mathbb{R}_{\geq 0}$. The objective is to find a matching $M \subseteq E$ that maximizes $v(M)$. In the online version, the set L arrives online. Once a vertex in L arrives, we get to know its incident edges. At any point in time, we know the values of the objective function only restricted to subsets of the edges incident to the vertices that have already arrived. This problem also generalizes the submodular matroid secretary problem with transversal matroids.

We present a 0.207α-competitive algorithm, where α could be $\frac{1}{3}$ for a simple greedy algorithm [28]. The best known approximation algorithms are local search algorithms that give a $\frac{1}{2+\epsilon}$-approximation on bipartite matchings [25, 12]. The previously best known online algorithm is the simulated greedy algorithm with a competitive ratio of $1/95$ [26].

Algorithm 2 first samples a pn-fraction of the input sequence for some constant p. Then, whenever a new candidate arrives, it α-approximates the optimal matching on the known part of the graph with respect to the submodular objective function. If the current online vertex is matched in this matching and if its matching partner is still available, then we add the pair to the output allocation. This design paradigm has been successfully applied to linear objective functions before [17]. However, in the submodular case, the individual contribution on an edge to the eventual objective function value depends on what other edges are selected. Using an approach similar to the one in the previous section, we keep dependencies manageable.

▶ **Theorem 10.** *Algorithm 2 for the submodular secretary matching problem is $\alpha(1-p^{1/p})(p^2 - O(\frac{1}{n}))$-competitive with sample size pn. For $p = 0.614$, the algorithm is 0.207α-competitive.*

We denote the set of matching edges allocated by the algorithm in rounds ℓ to n with $\mathrm{ALG}^{\geq \ell}$ and the set of tentative edges over the same period with $T^{\geq \ell}$. For $S, S' \subseteq E$, we denote the contribution of the subset S to S' by $v(S \mid S') = v(S \cup S') - v(S')$.

We show the following two lemmas.

▶ **Lemma 11.** *In every round ℓ fix the tentative edges that will be selected in the future rounds $\ell + 1, \ldots, n$. Then the marginal contribution of the tentative edge $e^{(\ell)}$ selected by the online algorithm in round ℓ is*

$$\mathbf{E}\left[v\left(\{e^{(\ell)}\} \;\middle|\; \mathrm{ALG}^{\geq \ell+1} \right) \;\middle|\; L^{\leq \ell}, T^{\geq \ell+1} \right] \geq \frac{1}{\ell} \left(v(\mathcal{A}(L^{\leq \ell})) - v(T^{\geq \ell+1}) \right) \quad .$$

Proof. We will use that $v\left(\{e^{(\ell)}\} \;\middle|\; \mathrm{ALG}^{\geq \ell+1}\right) \geq v\left(\{e^{(\ell)}\} \mid T^{\geq \ell+1}\right)$ because of submodularity of v and since $\mathrm{ALG}^{\geq \ell+1} \subseteq T^{\geq \ell+1}$. This allows us to avoid complex dependencies.

With $L^{\leq \ell}$ fixed, the algorithm's output $\mathcal{A}(L^{\leq \ell})$ is determined as well. The online vertex in round ℓ is as drawn uniformly at random from all vertices in $L^{\leq \ell}$. This gives us

$$\begin{aligned}\mathbf{E}\left[v\left(\{e^{(\ell)}\} \,\middle|\, T^{\geq \ell+1}\right) \,\middle|\, L^{\leq \ell}, T^{\geq \ell+1}\right] &\geq \frac{1}{\ell} v\left(\mathcal{A}(L^{\leq \ell}) \mid T^{\geq \ell+1}\right) \\ &\geq \frac{1}{\ell}\left(v(\mathcal{A}(L^{\leq \ell})) - v(T^{\geq \ell+1})\right) \quad . \qquad \blacktriangleleft\end{aligned}$$

This lemma is shown in a way similar to Proposition 2.

▶ **Lemma 12.** *The probability that a tentative edge $e^{(\ell)}$ is feasible given all vertices that arrived earlier $L^{\leq \ell}$ and all future tentative edges $T^{\geq \ell+1}$ is*

$$\mathbf{Pr}\left[\text{Accepted} \cup e^{(\ell)} \textit{ is a matching} \,\middle|\, L^{\leq \ell}, T^{\geq \ell+1}\right] \geq \frac{pn-1}{\ell-1} \quad .$$

This lemma was already shown in [17].

Proof of Theorem 10. Let $\hat{e}^{(\ell)} = \{e^{(\ell)}\}$ if Accepted$\cup e^{(\ell)}$ is a matching and empty otherwise. We combine Lemmas 11 and 12, and we get that in every round ℓ for a fixed set $L^{\leq \ell}$ and $T^{\geq \ell+1}$, we have

$$\mathbf{E}\left[v\left(\hat{e}^{(\ell)} \,\middle|\, \mathrm{ALG}^{\geq \ell+1}\right) \,\middle|\, L^{\leq \ell}, T^{\geq \ell+1}\right] \geq \frac{1}{\ell}\frac{pn-1}{\ell-1}\left(v(\mathcal{A}(L^{\leq \ell} \cup R)) - v(T^{\geq \ell+1})\right)$$

and therefore

$$\mathbf{E}\left[v\left(\hat{e}^{(\ell)} \,\middle|\, \mathrm{ALG}^{\geq \ell+1}\right)\right] \geq \frac{1}{\ell}\frac{pn-1}{\ell-1}\left(\mathbf{E}\left[v(\mathcal{A}(L^{\leq \ell} \cup R))\right] - \mathbf{E}\left[v(T^{\geq \ell+1})\right]\right) \quad .$$

We use Lemma 12 for each future tentative edge $e^{(\ell')} \in T^{\geq \ell+1}$ and upperbound $\ell' \leq n$. This gives us $\mathbf{E}\left[v(\mathrm{ALG}^{\geq \ell+1})\right] \geq p\mathbf{E}\left[v(T^{\geq \ell+1})\right]$. Furthermore, to bound $\mathbf{E}\left[v(\mathcal{A}(L^{\leq \ell} \cup R))\right]$, we use that the optimal solution on the subgraph induced by $L^{\leq \ell} \cup R$ is at least as good as the optimal solution restricted to the edges in this subgraph. As every edge appears with probability $\frac{\ell}{n}$ submodularity gives us $\mathbf{E}\left[v(\mathcal{A}(L^{\leq \ell} \cup R))\right] \geq \alpha \frac{\ell}{n} v(\mathrm{OPT})$. In combination with $\ell \geq pn$, this yields

$$\mathbf{E}\left[v\left(\hat{e}^{(\ell))} \,\middle|\, \mathrm{ALG}^{\geq \ell+1}\right)\right] \geq \frac{\alpha}{n}\frac{pn-1}{\ell-1} v(\mathrm{OPT}) - \frac{1}{\ell}\frac{1}{p}\mathbf{E}\left[v(\mathrm{ALG}^{\geq \ell+1})\right] \quad .$$

As $\mathrm{ALG}^{\geq \ell} = \hat{e}^{(\ell)} \cup \mathrm{ALG}^{\geq \ell+1}$, we get the following tail recursion

$$\begin{aligned}\mathbf{E}\left[v((\mathrm{ALG}^{\geq \ell})\right] &\geq \frac{\alpha}{n}\frac{pn-1}{\ell-1} v(\mathrm{OPT}) + \left(1 - \frac{1/p}{\ell}\right)\mathbf{E}\left[v(\mathrm{ALG}^{\geq \ell+1})\right] \\ &\geq \sum_{j=\ell}^{n}\prod_{i=\ell}^{j-1}\left(1 - \frac{1/p}{i}\right)\frac{1}{j-1}\left(p - \frac{1}{n}\right)\alpha v(\mathrm{OPT}) \quad .\end{aligned}$$

We use $\prod_{i=\ell}^{j-1}\left(1 - \frac{1/p}{i}\right) \geq \left(\frac{\ell - 1/p}{j - 1/p}\right)^{1/p}$, see Lemma 14 in Appendix B.1 for a proof. Additionally we use $\frac{1}{j-1} = \frac{1}{j-1/p}\frac{j-1/p}{j-1} = \frac{1}{j-1/p}\left(1 - \frac{1/p-1}{j-1}\right) \geq \frac{1}{j-1/p}\left(1 - \frac{1/p-1}{pn-1}\right)$ and get

$$\mathbf{E}\left[v(\mathrm{ALG}^{\geq \ell})\right] \geq \sum_{j=\ell}^{n}\frac{(\ell - 1/p)^{1/p}}{(j - 1/p)^{1/p+1}}\left(1 - \frac{1/p - 1}{pn-1}\right)\left(p - \frac{1}{n}\right)\alpha \mathrm{OPT} \quad .$$

We approximate the sum with the integral and get $\sum_{j=\ell}^{n} \frac{1}{(j-1/p)^{1/p+1}} \geq \int_{\ell}^{n} \frac{1}{(j-1/p)^{1/p+1}} dj - \frac{1}{(\ell-1-1/p)^{1/p+1}} = p\left(\frac{1}{(\ell-1/p)^{1/p}} - \frac{1}{(n-1/p)^{1/p}} - \frac{1}{(\ell-1-1/p)^{1/p+1}}\right)$. Together with $\frac{1}{n} = \frac{p-1/n}{pn-1}$ this gives us

$$\frac{\mathbf{E}\left[v(\mathrm{ALG}^{\geq \ell})\right]}{\mathrm{OPT}} \geq \alpha \left(1 - \left(\frac{\ell - 1/p}{n - 1/p}\right)^{1/p} - \frac{(\ell - 1/p)^{1/p}}{(\ell - 1 - 1/p)^{1/p+1}}\right)\left(p^2 - \frac{1+p^2-p-p/n}{pn-1}\right) .$$

Now the expected value of the online algorithm is $\mathbf{E}\left[v(\mathrm{ALG}^{\geq pn})\right]$. We have $\frac{pn-1/p}{n-1/p} = p\frac{n-1/p^2}{n-1/p} \leq p$ and $\frac{(\ell-1/p)^{1/p}}{(\ell-1-1/p)^{1/p+1}} = \left(1 + \frac{1}{\ell-1-1/p}\right)^{1/p} \frac{1}{\ell-1-1/p} \in O\left(\frac{1}{n}\right)$. This gives us

$$\mathbf{E}\left[v(\mathrm{ALG}^{\geq pn})\right] \geq \left(1 - p^{1/p}\right)\left(p^2 - O\left(\frac{1}{n}\right)\right) \alpha v(\mathrm{OPT}) \ . \qquad ◀$$

This bound on the expected competitive ratio has a local maximum of 0.207α when the parameter for the sample size is $p = 0.614$.

4 Submodular Function subject to Linear Packing Constraints

We now generalize the setting to feature arbitrary linear packing constraints. That is, each item j is associated a variable y_j and there are m constraints of the form $\sum_{j \in U} a_{i,j} y_j \leq b_i$ with $a_{i,j} \geq 0$. The coefficients $a_{i,j}$ are chosen by an adversary and are revealed to the online algorithm once the respective item arrives. Immediately and irrevocably, we have to either accept or reject the item, which corresponds to setting y_j to 0 or 1. The best previous result is a constant competitive algorithm for a single constraint and $\Omega(1/m)$-competitive for multiple constraints, where m is the number of constraints [4].

Our algorithms extend the ones presented in [18] from linear to submodular objective. Again, they rely on a suitable algorithm solving the offline optimization problem. In this case we need a fractional allocation $x \in [0,1]^U$, which we evaluate in terms of the multilinear extension $F(x) = \sum_{R \subseteq U} \left(\prod_{i \in R} f(R) x_i \prod_{i \notin R} (1 - x_i)\right)$. In more detail, we assume that for any packing polytope $P \subseteq [0,1]^U$, $F(\mathcal{A}_F(P)) \geq \alpha \sup_{x \in P} F(x)$. For example, the continuous greedy process by Calinescu et al. [7] provides a $(1 - 1/e)$-approximation in polynomial time. As the set P, we use $\mathcal{P}(\frac{\ell}{n}, S)$, which is defined to be the set of vectors $x \geq 0$, for which $Ax \leq \frac{\ell}{n} b$ and $x_i = 0$ if $i \notin S$. This is the polytope of the solution space with scaled down constraints and restricted on the variables that arrived so far.

Our bounds are parameterized in the capacity ratio B and the column sparsity d. The capacity ratio B is defined by $B = \min_{i \in [m]} \frac{b_i}{\max_{j \in [n]} a_{i,j}}$. The column sparsity d is the maximal number of non-zero entries in a column of the constraint matrix A. We consider two variants of this problem, where either the B and d are known to the algorithm or not.

▶ **Theorem 13.** *There is an $\Omega\left(\alpha d^{-\frac{2}{B-1}}\right)$-competitive online algorithm for submodular maximization subject to linear constraints with unknown capacity ratio $B \geq 2$ and unknown column sparsity d. This improves to $\Omega\left(\alpha d^{-\frac{1}{B-1}}\right)$ if B and d are known.*

Note that, although the algorithm $\mathcal{A}$ returns fractional solutions, the output of our online algorithms is integral. The competitive ratio is between the integral solution of the online algorithm and the optimal fractional allocation with respect to the multilinear extension.

The proof for Theorem 13 combines ideas from Section 2 and 3 with [18]. Due to space limitations, the details are only included in the full version.

Algorithm 3: Submodular Function Maximization subject to Linear Constraints

```
Let $x := 0$ and $S := \emptyset$ be the index set of known requests;
for each arriving request $j$ do                                          // steps $\ell = 1$ to $n$
    Set $S := S \cup \{j\}$ and $\ell := |S|$;
    Let $\tilde{x}^{(\ell)} := \mathcal{A}_F(\mathcal{P}(\frac{\ell}{n}, S))$;        // fractional $\alpha$-approximation on scaled
     polytope
    Set $\hat{x}_j^{(\ell)} = 1$ with probability $\tilde{x}_j^{(\ell)}$;      // tentative allocation after rand.
     rounding
    if $A(x + \hat{x}^{(\ell)}) \leq b$ then                                // feasibility test
        Set $x^{(\ell)} := \hat{x}^{(\ell)}$, $x := x + \hat{x}^{(\ell)}$;                   // online allocation
```

References

1 Shipra Agrawal and Nikhil R. Devanur. Fast algorithms for online stochastic convex programming. In *Proc. 26th Symp. Discr. Algorithms (SODA)*, pages 1405–1424, 2015. doi:10.1137/1.9781611973730.93.

2 Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. A dynamic near-optimal algorithm for online linear programming. *Operations Research*, 62(4):876–890, 2014. doi:10.1287/opre.2014.1289.

3 Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, secretary problems, and online mechanisms. In *Proc. 18th Symp. Discr. Algorithms (SODA)*, pages 434–443, 2007. URL: http://dl.acm.org/citation.cfm?id=1283383.1283429.

4 MohammadHossein Bateni, Mohammad Taghi Hajiaghayi, and Morteza Zadimoghaddam. Submodular secretary problem and extensions. *ACM Trans. Algorithms*, 9(4):32, 2013. doi:10.1145/2500121.

5 Niv Buchbinder and Moran Feldman. Constrained submodular maximization via a non-symmetric technique. *CoRR*, abs/1611.03253, 2016. URL: http://arxiv.org/abs/1611.03253.

6 Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In *Proc. 25th Symp. Discr. Algorithms (SODA)*, pages 1433–1452, 2014. doi:10.1137/1.9781611973402.106.

7 Gruia Călinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.*, 40(6):1740–1766, 2011. doi:10.1137/080733991.

8 Nikhil R. Devenur and Thomas P. Hayes. The adwords problem: online keyword matching with budgeted bidders under random permutations. In *Proc. 10th Conf. Econom. Comput. (EC)*, pages 71–78, 2009. doi:10.1145/1566374.1566384.

9 Alina Ene and Huy L. Nguyen. Constrained submodular maximization: Beyond 1/e. In *Proc. 57th Symp. Foundations of Computer Science (FOCS)*, pages 248–257, 2016. doi:10.1109/FOCS.2016.34.

10 Moran Feldman and Rani Izsak. Building a good team: Secretary problems and the supermodular degree. In *Proc. 28th Symp. Discr. Algorithms (SODA)*, pages 1651–1670, 2017. doi:10.1137/1.9781611974782.109.

11 Moran Feldman, Joseph Naor, and Roy Schwartz. Improved competitive ratios for submodular secretary problems (extended abstract). In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques – 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA,*

August 17-19, 2011. Proceedings, pages 218–229, 2011. doi:10.1007/978-3-642-22935-0_19.

12 Moran Feldman, Joseph Naor, Roy Schwartz, and Justin Ward. Improved approximations for k-exchange systems – (extended abstract). In *Proc. 19th European Symp. Algorithms (ESA)*, pages 784–798, 2011. doi:10.1007/978-3-642-23719-5_66.

13 Moran Feldman, Ola Svensson, and Rico Zenklusen. A simple $O(\log\log(\text{rank}))$-competitive algorithm for the matroid secretary problem. In *Proc. 26th Symp. Discr. Algorithms (SODA)*, pages 1189–1201, 2015. doi:10.1137/1.9781611973730.79.

14 Moran Feldman and Rico Zenklusen. The submodular secretary problem goes linear. In *Proc. 56th Symp. Foundations of Computer Science (FOCS)*, pages 486–505, 2015. doi:10.1109/FOCS.2015.37.

15 Anupam Gupta, Aaron Roth, Grant Schoenebeck, and Kunal Talwar. Constrained non-monotone submodular maximization: Offline and secretary algorithms. In *Proc. 6th Int'l Conf. Web and Internet Economics (WINE)*, pages 246–257, 2010. doi:10.1007/978-3-642-17572-5_20.

16 Michael Kapralov, Ian Post, and Jan Vondrák. Online submodular welfare maximization: Greedy is optimal. In *Proc. 24th Symp. Discr. Algorithms (SODA)*, pages 1216–1225, 2013. doi:10.1137/1.9781611973105.88.

17 Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. An optimal online algorithm for weighted bipartite matching and extensions to combinatorial auctions. In *Proc. 21st European Symp. Algorithms (ESA)*, pages 589–600, 2013. doi:10.1007/978-3-642-40450-4_50.

18 Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. Primal beats dual on online packing lps in the random-order model. In *Proc. 46th Symp. Theory of Computing (STOC)*, pages 303–312, 2014. doi:10.1145/2591796.2591810.

19 Robert D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Proc. 16th Symp. Discr. Algorithms (SODA)*, pages 630–631, 2005. URL: http://dl.acm.org/citation.cfm?id=1070432.1070519.

20 Nitish Korula, Vahab S. Mirrokni, and Morteza Zadimoghaddam. Online submodular welfare maximization: Greedy beats 1/2 in random order. In *Proc. 47th Symp. Theory of Computing (STOC)*, pages 889–898, 2015. doi:10.1145/2746539.2746626.

21 Nitish Korula and Martin Pál. Algorithms for secretary problems on graphs and hypergraphs. In *Proc. 36th Int'l Coll. Autom. Lang. Program. (ICALP)*, pages 508–520, 2009. doi:10.1007/978-3-642-02930-1_42.

22 Andreas Krause and Daniel Gloving. Submodular function maximization. In *Tractability: Practical Approaches to Hard Problems*, chapter 3. Cambridge University Press, 2014.

23 Ariel Kulik, Hadas Shachnai, and Tami Tamir. Approximations for monotone and nonmonotone submodular maximization with knapsack constraints. *Math. Oper. Res.*, 38(4):729–739, 2013. doi:10.1287/moor.2013.0592.

24 Oded Lachish. O(log log rank) competitive ratio for the matroid secretary problem. In *Proc. 55th Symp. Foundations of Computer Science (FOCS)*, pages 326–335, 2014. doi:10.1109/FOCS.2014.42.

25 Jon Lee, Maxim Sviridenko, and Jan Vondrák. Submodular maximization over multiple matroids via generalized exchange properties. *Math. Oper. Res.*, 35(4):795–806, 2010. doi:10.1287/moor.1100.0463.

26 Tengyu Ma, Bo Tang, and Yajun Wang. The simulated greedy algorithm for several submodular matroid secretary problems. *Theoret. Comput. Sci.*, 58(4):681–706, 2016. doi:10.1007/s00224-015-9642-4.

27 Marco Molinaro and R. Ravi. The geometry of online packing linear programs. *Math. Oper. Res.*, 39(1):46–59, 2014. doi:10.1287/moor.2013.0612.

28 George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions – II. *Math. Prog.*, 14(1):265–294, 1978. doi:10.1007/BF01588971.

29 G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Math. Oper. Res.*, 3(3):177–188, 1978. doi:10.1287/moor.3.3.177.

30 Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.*, 32(1):41–43, 2004. doi:10.1016/S0167-6377(03)00062-2.

A Missing Details in Section 2

A.1 Continued Proof of Lemma 4

To show the lemma, we perform an induction on r. Note that Equation (2) trivially holds for $r = 0$. In order to prove it holds for a given $r > 0$, we assume that it is fulfilled for $r - 1$ for all $\ell \in [n]$. From this, we will conclude that Equation (2) also holds for r for all $\ell \in [n]$. To show that (3) is solved by (2), we use the induction hypothesis and plug in the bound for $\mathbf{E}\left[v(\mathrm{ALG}_{r-1}^{\geq j+1})\right]$. This gives us

$$\begin{aligned}\frac{\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right]}{\alpha v(\mathrm{OPT})} &\geq \sum_{j=\ell}^{n} \left(\frac{\ell-k}{j-k}\right)^k \frac{k-1}{j+1}\left(\frac{(r-1)(j+1)}{(k-1)n} - \frac{3k^2(r-1)}{(k-1)n} + \frac{1}{n}\right.\\ &\qquad \left. - \frac{1}{k-1}\left(\frac{j+1}{n}\right)^k \sum_{r'=0}^{r-2}\sum_{i=0}^{r'} \frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{j+1}\right)\right)\\ &= \sum_{j=\ell}^{n}\left(\frac{\ell-k}{j-k}\right)^k \frac{r}{n} - \sum_{j=\ell}^{n}\left(\frac{\ell-k}{j-k}\right)^k \frac{3k^2(r-1)}{(j+1)n}\\ &\qquad - \sum_{j=\ell}^{n}\left(\frac{\ell-k}{j-k}\right)^k \frac{1}{j+1}\left(\frac{j+1}{n}\right)^k \sum_{r'=0}^{r-2}\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{j+1}\right) \quad .\end{aligned}$$

In the negative terms, we bound $\frac{\ell-k}{j-k} \leq \frac{\ell}{j}$ and use $\left(\frac{j+1}{j}\right)^k \leq e^{\frac{k}{j}} \leq e^{\frac{k}{\ell}} \leq 1 + 2\frac{k}{\ell}$. Finally in the last sum, we bound $\frac{1}{j+1} \leq \frac{1}{\ell}$ once

$$\begin{aligned}\frac{\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right]}{\alpha v(\mathrm{OPT})} &\geq \sum_{j=\ell}^{n}\left(\frac{\ell-k}{j-k}\right)^k \frac{r}{n} - \sum_{j=\ell}^{n}\left(\frac{\ell}{j}\right)^k \frac{3k^2(r-1)}{\ell n}\\ &\qquad - \left(\frac{\ell}{n}\right)^k \sum_{j=\ell}^{n} \frac{(1+2\frac{k}{\ell})}{j+1}\sum_{r'=0}^{r-2}\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{j+1}\right) \quad .\end{aligned}$$

We approximate both sums over j through integrals by using

$$\sum_{j=\ell}^{n}\frac{1}{(j-k)^k} \geq \int_{\ell}^{n}\frac{1}{(j-k)^k}dj = \frac{1}{k-1}\left(\frac{1}{(\ell-k)^{k-1}} - \frac{1}{(n-k)^{k-1}}\right)$$

and

$$\sum_{j=\ell}^{n}\frac{\ln^i({}^{n}/_{(j+1)})}{j+1} \leq \int_{\ell-1}^{n-1}\frac{\ln^i({}^{n}/_{(j+1)})}{j+1}dj = \left[-\frac{\ln^{i+1}({}^{n}/_{(j+1)})}{i+1}\right]_{\ell-1}^{n-1} = \frac{\ln^{i+1}({}^{n}/_{\ell})}{i+1} \quad .$$

This yields

$$\begin{aligned}\frac{\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right]}{\alpha v(\mathrm{OPT})} &\geq \frac{r(\ell-k)}{(k-1)n}\left(1-\left(\frac{\ell-k}{n-k}\right)^{k-1}\right) - \frac{3k^2(r-1)}{(k-1)n}\left(1-\left(\frac{\ell}{n}\right)^{k-1}\right) \\ &\quad - \left(\frac{\ell}{n}\right)^k\left(1+2\frac{k}{\ell}\right)\sum_{r'=0}^{r-2}\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\frac{\ln^{i+1}\left(\frac{n}{\ell}\right)}{i+1} \quad .\end{aligned}$$

We perform an index shift in the inner sum and propagate the shift to the outer sum

$$\begin{aligned}\sum_{r'=0}^{r-2}\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\frac{\ln(n/\ell)^{i+1}}{i+1} &= \frac{1}{k-1}\sum_{r'=0}^{r-2}\sum_{i=1}^{r'+1}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{\ell}\right) \\ &= \frac{1}{k-1}\sum_{r'=1}^{r-1}\sum_{i=1}^{r'}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{\ell}\right) \\ &= \frac{1}{k-1}\sum_{r'=0}^{r-1}\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{\ell}\right) - \frac{r}{k-1} \quad .\end{aligned}$$

Now we solve the brackets and use the term split off in the index shift to simplify the expression. We get

$$\begin{aligned}\frac{\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right]}{\alpha v(\mathrm{OPT})} &\geq \frac{r(\ell-k)}{(k-1)n} - \frac{r(\ell-k)}{(k-1)n}\left(\frac{\ell-k}{n-k}\right)^{k-1} + \left(\frac{\ell}{n}\right)^k\frac{\left(1+2\frac{k}{\ell}\right)}{k-1}r \\ &\quad - \left(\frac{\ell}{n}\right)^k\frac{\left(1+2\frac{k}{\ell}\right)}{k-1}\sum_{r'=0}^{r-1}\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{\ell}\right) - \frac{3k^2(r-1)}{(k-1)n} \\ &\geq \frac{r\ell}{(k-1)n} - \frac{rk}{(k-1)n} - \left(\frac{\ell}{n}\right)^k\frac{\left(1+2\frac{k}{\ell}\right)}{k-1}\sum_{r'=0}^{r-1}\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{\ell}\right) \\ &\quad - \frac{3k^2(r-1)}{(k-1)n} \quad .\end{aligned}$$

At this point, we only have to show that the following inequality holds

$$\frac{rk}{(k-1)n} + \left(\frac{\ell}{n}\right)^k\frac{2\frac{k}{\ell}}{k-1}\sum_{r'=0}^{r-1}\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{\ell}\right) + \frac{3k^2(r-1)}{(k-1)n} \leq \frac{3k^2r}{(k-1)n} \quad .$$

We bound the inner sum with the corresponding exponential function

$$\sum_{i=0}^{r'}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{\ell}\right) \leq \sum_{i=0}^{\infty}\frac{(k-1)^i}{i!}\ln^i\left(\frac{n}{\ell}\right) = \exp\left((k-1)\ln\left(\frac{n}{\ell}\right)\right) = \left(\frac{n}{\ell}\right)^{k-1} \quad .$$

This term is independent of r'. We eliminate the sum over r' and get

$$\frac{rk}{(k-1)n} + \frac{\ell}{n}\frac{r2\frac{k}{\ell}}{k-1} = \frac{3kr}{(k-1)n} \leq \frac{3k^2}{(k-1)n} \quad .$$

A.2 Detailed Proof of Theorem 1

To complete the proof of the theorem, we apply Lemma 4 for $\ell = pn$ and $r = k$. This gives us $\mathbf{E}\left[v(\mathrm{ALG})\right] = \mathbf{E}\left[v(\mathrm{ALG}_k^{\geq pn})\right]$ and thus

$$\mathbf{E}\left[v(\mathrm{ALG})\right] \geq \left(\frac{pk}{k-1} - \frac{1}{k-1} p^k \sum_{r'=0}^{k-1} \sum_{i=0}^{r'} \frac{(k-1)^i}{i!} \ln^i\left(\frac{1}{p}\right) - \frac{6k^2}{n} \right) \cdot \alpha v(\mathrm{OPT}) \ .$$

For p such that $pn = \lceil \frac{n}{e} \rceil$, we have $\frac{1}{e} \leq p \leq \frac{1}{e} + \frac{1}{n}$ and $\ln\left(\frac{1}{p}\right) = 1 + \ln\left(\frac{n}{n+e}\right) \leq 1$. This allows us to reorder the occurring double sum as follows

$$\begin{aligned} \sum_{r'=0}^{k-1} \sum_{i=0}^{r'} \frac{(k-1)^i}{i!} &= \sum_{i=0}^{k-1} (k-i) \frac{(k-1)^i}{i!} = k \sum_{i=0}^{k-1} \frac{(k-1)^i}{i!} - (k-1) \sum_{i=1}^{k-1} \frac{(k-1)^{i-1}}{(i-1)!} \\ &= \sum_{i=0}^{k-1} \frac{(k-1)^i}{i!} + \frac{(k-1)^k}{(k-1)!} \ . \end{aligned}$$

By definition of the exponential function $e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!}$. For $x > 0$, all terms of the infinite sum are positive. This yields $e^x \geq \sum_{i=0}^{k-1} \frac{x^i}{i!} + \frac{x^k}{k!} + \frac{x^{k+1}}{(k+1)!}$ and thus by setting $x = k-1$ we get

$$\sum_{r'=0}^{k-1} \sum_{i=0}^{r'} \frac{(k-1)^i}{i!} \leq e^{k-1} - \frac{(k-1)^k}{k!} - \frac{(k-1)^{k+1}}{(k+1)!} + \frac{(k-1)^k}{(k-1)!} \ .$$

We have $p^k e^{k-1} \leq \left(\frac{1}{e} + \frac{1}{n}\right)^k e^{k-1} = \left(1 + \frac{e}{n}\right)^{k-1} \left(\frac{1}{e} + \frac{1}{n}\right) \leq e^{\frac{ek}{n}} \left(\frac{1}{e} + \frac{1}{n}\right)$, this implies

$$\begin{aligned} \frac{\mathbf{E}\left[v(\mathrm{ALG})\right]}{\alpha v(\mathrm{OPT})} &\geq \frac{k}{e(k-1)} - \frac{\left(\frac{1}{e} + \frac{1}{n}\right)^k}{(k-1)} \left(e^{k-1} - \frac{(k-1)^k}{k!} - \frac{(k-1)^{k+1}}{(k+1)!} + \frac{(k-1)^k}{(k-1)!} \right) - \frac{6k^2}{n} \\ &= \frac{k}{e(k-1)} - \frac{e^{\frac{ek}{n}}}{e(k-1)} - \frac{e^{\frac{ek}{n}}}{n(k-1)} \\ &\quad + \left(\frac{1}{e} + \frac{1}{n}\right)^k \left(\frac{(k-1)^{k-1}}{k!} + \frac{(k-1)^k}{(k+1)!} - \frac{(k-1)^{k-1}}{(k-1)!} \right) - \frac{6k^2}{n} \\ &= \frac{k - e^{\frac{ek}{n}}}{e(k-1)} - \left(\frac{1}{e} + \frac{1}{n}\right)^k \frac{k-1}{k+1} \frac{(k-1)^{k-1}}{(k-1)!} - \frac{6k^2}{n} \ . \end{aligned}$$

At this point, we apply the Stirling approximation $(k-1)! \geq \sqrt{2\pi(k-1)} \left(\frac{k-1}{e}\right)^{k-1}$ and get

$$\begin{aligned} \frac{\mathbf{E}\left[v(\mathrm{ALG})\right]}{\alpha v(\mathrm{OPT})} &\geq \frac{1}{e} - \frac{e^{\frac{ek}{n}} - 1}{e(k-1)} - \left(\frac{1}{e} + \frac{1}{n}\right)^k e^{k-1} \frac{\sqrt{k-1}}{(k+1)\sqrt{2\pi}} - \frac{6k^2}{n} \\ &= \frac{1}{e} - \frac{e^{\frac{ek}{n}} - 1}{e(k-1)} - e^{\frac{ek}{n}} \left(\frac{1}{e} + \frac{1}{n}\right) \frac{\sqrt{k-1}}{(k+1)\sqrt{2\pi}} - \frac{6k^2}{n} \ . \end{aligned}$$

For every fixed k, we can assume that n is arbitrarily larger. This can be guaranteed, for example, through dummy elements with marginal gain zero for all sets. In the limit, this yields

$$\frac{\mathbf{E}\left[v(\mathrm{ALG})\right]}{\alpha v(\mathrm{OPT})} \geq \frac{1}{e} \left(1 - \frac{\sqrt{k-1}}{(k+1)\sqrt{2\pi}}\right) \ .$$

A.3 Proof of Claim 7

Proof. We perform an induction on ℓ. Assume that the claim has been shown for all r for $\ell+1$. In Lemma 3, we have shown

$$\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq \frac{1}{\ell}\left(\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right] + (k-1)\mathbf{E}\left[v(\mathrm{ALG}_{r-1}^{\geq \ell+1})\right] + (\ell-k)\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell+1})\right]\right).$$

Now we use the induction hypothesis

$$\begin{aligned}\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq{}& \frac{1}{\ell}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\\ &+ \frac{k-1}{\ell}\sum_{j=\ell+1}^{n}\frac{a_{\ell+1,j-1}}{j}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\sum_{r'=0}^{r-2}\sum_{\substack{M\subseteq\{\ell+1,\ldots,j-1\}\\|M|=r'}}\left(\prod_{i\in M}\frac{k-1}{i}\right)\\ &+ \frac{\ell-k}{\ell}\sum_{j=\ell+1}^{n}\frac{a_{\ell+1,j-1}}{j}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\sum_{r'=0}^{r-1}\sum_{\substack{M\subseteq\{\ell+1,\ldots,j-1\}\\|M|=r'}}\left(\prod_{i\in M}\frac{k-1}{i}\right).\end{aligned}$$

We perform an index shift, use $\frac{\ell-k}{\ell}a_{\ell+1,j-1} = a_{\ell,j-1}$ and get

$$\begin{aligned}\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] ={}& \frac{a_{\ell,\ell-1}}{\ell}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\\ &+ \sum_{j=\ell+1}^{n}\frac{a_{\ell+1,j-1}}{j}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\sum_{r'=1}^{r-1}\frac{k-1}{\ell}\sum_{\substack{M\subseteq\{\ell+1,\ldots,j-1\}\\|M|=r'-1}}\left(\prod_{i\in M}\frac{k-1}{i}\right)\\ &+ \sum_{j=\ell+1}^{n}\frac{a_{\ell,j-1}}{j}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\sum_{r'=0}^{r-1}\sum_{\substack{M\subseteq\{\ell+1,\ldots,j-1\}\\|M|=r'}}\left(\prod_{i\in M}\frac{k-1}{i}\right).\end{aligned}$$

We have $\frac{k-1}{\ell} \geq \frac{k-1}{i}$ for all $i \geq \ell$ and therefore we can merge the factor for the current round into the product. In a sense the $\frac{k-1}{\ell}$ factor stands for choosing an item in the current round, and it gets worse if we chose one in a future round instead. Additionally we use $a_{\ell+1,j-1} \geq a_{\ell,j-1}$ and omit the second large sum entirely.

For the final equality we use the fact that $\sum_{r'=0}^{r-1}\sum_{\substack{M\subseteq\emptyset\\|M|=r'}}\left(\prod_{i\in M}\frac{k-1}{i}\right) = 1$ because the inner sum is empty for all $r' > 0$

$$\begin{aligned}\mathbf{E}\left[v(\mathrm{ALG}_r^{\geq \ell})\right] \geq{}& \frac{a_{\ell,\ell-1}}{\ell}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\\ &+ \sum_{j=\ell+1}^{n}\frac{a_{\ell,j-1}}{j}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\sum_{r'=0}^{r-1}\sum_{\substack{M\subseteq\{\ell,\ldots,j-1\}\\|M|=r'}}\left(\prod_{i\in M}\frac{k-1}{i}\right)\\ ={}& \sum_{j=\ell}^{n}\frac{a_{\ell,j-1}}{j}\mathbf{E}\left[v(\mathcal{A}(U^{\leq \ell}))\right]\sum_{r'=0}^{r-1}\sum_{\substack{M\subseteq\{\ell,\ldots,j-1\}\\|M|=r'}}\left(\prod_{i\in M}\frac{k-1}{i}\right).\end{aligned}$$ ◀

A.4 Proof of Claim 8

Proof. Towards a proof, we show that $t_{\ell,j+1} \leq \beta_j t_{\ell,j}$ for some $\beta_j \leq 1$. We consider the definition of $t_{\ell,j+1}$ and split of a double sum that contains all terms where $j \in M$. In those terms, we know that j is selected and therefore the factor $\frac{k-1}{j}$ should always exist in the product. We get

$$t_{\ell,j+1} = a_{\ell,j} \sum_{r'=0}^{r-1} \sum_{\substack{M\subseteq\{\ell,\dots,j\}\\|M|=r'}} \left(\prod_{i\in M} \frac{k-1}{i}\right)$$

$$= a_{\ell,j} \left(\sum_{r'=0}^{r-1} \sum_{\substack{M\subseteq\{\ell,\dots,j-1\}\\|M|=r'}} \left(\prod_{i\in M} \frac{k-1}{i}\right) + \frac{k-1}{j} \sum_{r'=0}^{r-1} \sum_{\substack{M\subseteq\{\ell,\dots,j-1\}\\|M|=r'-1}} \left(\prod_{i\in M} \frac{k-1}{i}\right) \right) .$$

Both double sums are nearly identical. We fill up the missing terms in the smaller one and bound by the following expression. Finally, we replace the remaining double sum with the definition of $t_{\ell,j}$

$$t_{\ell,j+1} \le a_{\ell,j} \left(1 + \frac{k-1}{j}\right) \sum_{r'=0}^{r-1} \sum_{\substack{M\subseteq\{\ell,\dots,j-1\}\\|M|=r'}} \left(\prod_{i\in M} \frac{k-1}{i}\right) = \frac{a_{\ell,j}}{a_{\ell,j-1}} \left(1 + \frac{k-1}{j}\right) t_{\ell,j} .$$

As we have $\frac{a_{\ell,j}}{a_{\ell,j-1}} \left(1 + \frac{k-1}{j}\right) = \left(1 + \frac{k-1}{j}\right)\left(1 - \frac{k}{j}\right) = 1 - \frac{k}{j} + \frac{k-1}{j} - \frac{k(k-1)}{j^2} \le 1$ the claim follows. ◀

B Missing Details in Section 3: Submodular Matching

B.1 Missing Details in the Proof of Theorem 10: Competitive Ratio for Submodular Matching

In the proof of Theorem 10, we also required the following technical lemma that is not problem-specific.

▶ **Lemma 14.** *For $i > c \geq 1$, we have*

$$\prod_{i=j}^{k} \left(1 - \frac{c}{i}\right) \geq \left(\frac{j-c}{k-c+1}\right)^c .$$

Proof. As first step, we show that

$$1 - \frac{c}{i} = \frac{i-c}{i} \geq \left(\frac{i-c}{i-c+1}\right)^c = \left(1 - \frac{1}{i-c+1}\right)^c .$$

This is equivalent to

$$\frac{i-c}{(i-c)^c} \geq \frac{i}{(i-c+1)^c} .$$

Now we show that this inequality holds for all $i > c \geq 1$. We define the function $f : [0,1] \to \mathbb{R}$ such that

$$f(x) = \frac{i-cx}{(i-c+1)^c} .$$

This function has the properties that $f(0) = \frac{i}{(i-c+1)^c}$ and $f(1) = \frac{i-c}{(i-c)^c}$. We show that f is non-decreasing increasing and therefore the inequality holds as well. The derivative f' of f is

$$f'(x) = \frac{-c(i-c+1-x)^c - (i-cx)c(i-c+1-x)^{(c-1)}(-1)}{(i-c+1-x)^{2c}} .$$

It suffice to show that f' is non-negative for all $x \in [0,1]$. This holds true if the numerator is positive for all $x \in [0,1]$ because the denominator is guaranteed to be positive with $i > c$ and $x \in [0,1]$. We have

$$\begin{aligned} -c(i-c+1-x)-(i-cx)c(-1) &\geq 0 \\ c(i-cx) &\geq c(i-c+1-x) \\ c-1 &\geq (c-1)x \ . \end{aligned}$$

This directly gives us the proof for the lemma

$$\prod_{i=j}^{k}\left(1-\frac{c}{i}\right) \geq \prod_{i=j}^{k}\left(1-\frac{1}{i-c+1}\right)^{c} = \left(\prod_{i=j}^{k}\frac{i-c}{i-c+1}\right)^{c} = \left(\frac{j-c}{k-c+1}\right)^{c} \ . \qquad ◀$$

On the Integrality Gap of the Prize-Collecting Steiner Forest LP*

Jochen Könemann[1], Neil Olver[2], Kanstantsin Pashkovich[3], R. Ravi[4], Chaitanya Swamy[5], and Jens Vygen[6]

1 **Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada**
jochen@uwaterloo.ca
2 **Department of Econometrics and Operations Research, Vrije Universiteit Amsterdam, Amsterdam, The Netherlands; and CWI, Amsterdam, The Netherlands**
n.olver@vu.nl
3 **Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada**
kpashkovich@uwaterloo.ca
4 **Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, USA**
ravi@andrew.cmu.edu
5 **Department of Combinatorics and Optimization, University of Waterloo, Waterloo, ON, Canada**
jcswamy@uwaterloo.ca
6 **Research Institute for Discrete Mathematics, Universität Bonn, Bonn, Germany**
vygen@or.uni-bonn.de

Abstract

In the *prize-collecting Steiner forest* (**PCSF**) problem, we are given an undirected graph $G = (V, E)$, edge costs $\{c_e \geq 0\}_{e \in E}$, terminal pairs $\{(s_i, t_i)\}_{i=1}^k$, and penalties $\{\pi_i\}_{i=1}^k$ for each terminal pair; the goal is to find a forest F to minimize $c(F) + \sum_{i:(s_i,t_i) \text{ not connected in } F} \pi_i$. The *Steiner forest* problem can be viewed as the special case where $\pi_i = \infty$ for all i. It was widely believed that the integrality gap of the natural (and well-studied) linear-programming (LP) relaxation for **PCSF** (PCSF-LP) is at most 2. We dispel this belief by showing that the integrality gap of this LP is at least 9/4. This holds even for planar graphs. We also show that using this LP, one cannot devise a Lagrangian-multiplier-preserving (LMP) algorithm with approximation guarantee better than 4. Our results thus show a separation between the integrality gaps of the LP-relaxations for prize-collecting and non-prize-collecting (i.e., standard) Steiner forest, as well as the approximation ratios achievable relative to the optimal LP solution by LMP- and non-LMP- approximation algorithms for **PCSF**. For the special case of *prize-collecting Steiner tree* (**PCST**), we prove that the natural LP relaxation admits basic feasible solutions with all coordinates of value at most 1/3 and all edge variables positive. Thus, we rule out the possibility of approximating **PCST** with guarantee better than 3 using a direct iterative rounding method.

1998 ACM Subject Classification G.1.6 [Optimization] Integer Programming, Linear Programming

Keywords and phrases Integrality gap, Steiner tree, Steiner forest, prize-collecting, Lagrangian-multiplier-preserving

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.17

* NO was partially supported by an NWO Veni grant. RR was supported in part by the U.S. National Science Foundation under award number CCF-1527032. CS was supported in part by NSERC grant 327620-09 and an NSERC Discovery Accelerator Supplement Award.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 17; pp. 17:1–17:13
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction and Background

In an instance of the well-studied Steiner tree problem one is given an undirected graph $G = (V, E)$, a non-negative cost c_e for each edge $e \in E$, and a set of *terminals* $R \subseteq V$. The goal is to find a minimum-cost tree in G spanning R. In the more general *Steiner forest* problem, terminals are replaced by *terminal pairs* $(s_1, t_1), \dots, (s_k, t_k)$ and the goal now becomes to compute a minimum-cost forest that connects s_i to t_i for all i. Both of the above problems are well-known to be NP- and APX-hard [7, 17]. The best-known approximation algorithm for the Steiner tree problem is due to Byrka et al. [5] (see also [11]) and achieves an approximation ratio of $\ln 4 + \epsilon$, for any $\epsilon > 0$; the Steiner forest problem admits a $(2 - 1/k)$-approximation algorithm [1, 12].

Our work focuses on the *prize-collecting* versions of the above problems. In the prize-collecting Steiner tree problem (**PCST**) we are given a Steiner-tree instance and a non-negative penalty π_v for each terminal $v \in R$. The goal is to find a tree T that minimizes $c(T) + \pi(T)$, where $c(T)$ denotes the total cost of all edges in T, and $\pi(T)$ denotes the total penalty of all terminals not spanned by T. In the prize-collecting Steiner forest problem (**PCSF**), we are given a Steiner-forest instance and a non-negative penalty π_i for each terminal pair (s_i, t_i), and the goal is to find forest F that minimizes $c(F) + \pi(F)$ where, similar to before, $c(F)$ is the total cost of forest F, and $\pi(F)$ denotes the total penalty of terminal pairs that are not connected by F. We can view **PCST** as a special case of **PCSF** by guessing a node r in the optimal tree, and then modeling each vertex in $v \in R \setminus \{r\}$ by the terminal pair (v, r).

The natural integer program (IP) for **PCSF** (see e.g. [3]) uses a binary variable x_e for every edge $e \in E$ whose value is 1 if e is part of the forest corresponding to x. The IP also has a variable z_i for each pair (s_i, t_i) whose value is 1 if s_i and t_i are *not* connected by the forest corresponding to x. We use $i \odot S$ for the predicate that is *true* if $S \subseteq V$ contains exactly one of s_i and t_i, and *false* otherwise. We use $\delta(S)$ to denote the set of edges with exactly one endpoint in S. In any integer solution to the LP relaxation below, the constraints insist that every cut separating pair (s_i, t_i) must be crossed by the forest unless we set z_i to 1 and pay the penalty for not connecting the terminals.

$$\begin{aligned} \min \quad & c^\top x + \pi^\top z & & \text{(PCSF-LP)} \\ \text{s.t.} \quad & x(\delta(S)) + z_i \geq 1 & \forall S \subseteq V,\ i \odot S & \\ & x, z \geq \mathbb{0}. & & \end{aligned}$$

Bienstock et al. [3] first presented a 3-approximation for **PCST** via a natural *threshold rounding* technique applied to this LP relaxation. This idea also works for **PCSF**, and proceeds as follows. First, we compute a solution (x, z) to the above LP. Let R' be the set of terminal pairs (s_i, t_i) with $z_i < 1/3$. Note that $\frac{3}{2} \cdot x$ is a feasible solution for the standard Steiner-forest cut-based LP (obtained from (PCSF-LP) by deleting the z variables) on the instance restricted to R'. Thus, applying an LP-based 2-approximation for Steiner forest [1, 12] to terminal pairs R' yields a forest F' of cost at most $2 \cdot \frac{3}{2} c^\top x = 3c^\top x$. The total penalty of the disconnected pairs is at most $3 \cdot \pi^\top z$. Hence, $c(F') + \pi(F')$ is bounded by $3(c^\top x + \pi^\top z)$, and the algorithm is a 3-approximation. Goemans showed that by choosing a random threshold (instead of the value 1/3) from a suitable distribution, one can obtain an improved performance guarantee of $1/(1 - e^{-1/2}) \approx 2.5415$ (see page 136 of [20], which attributes the corresponding randomized algorithm for PCST in Section 5.7 of [20] to Goemans).

Goemans and Williamson [12] later presented a primal-dual 2-approximation for **PCST** based on the Steiner tree special case (PCST-LP) of (PCSF-LP). In fact, the algorithm gives even a slightly better guarantee; it produces a tree T such that

$$c(T) + 2\pi(T) \le 2 \cdot \texttt{opt}_{\text{PCST-LP}},$$

where $\texttt{opt}_{\mathrm{PCST-LP}}$ is the optimum value of (PCST-LP). Algorithms for prize-collecting problems that achieve a performance guarantee of the form

$$c(F) + \beta \cdot \pi(F) \le \beta \cdot \texttt{opt}$$

are called *β-Lagrangian-multiplier preserving* (β-LMP) algorithms. Such algorithms are useful, for instance, for obtaining approximation algorithms for the *partial* covering version of the problem, which in the case of Steiner tree and Steiner forest translates to connecting at least a desired number of terminals (e.g., see [4, 16, 8, 9, 18]). Archer et al. [2] later used the strengthened guarantee of Goemans and Williamson's LMP algorithm for **PCST** to obtain a 1.9672-approximation algorithm for the problem.

The best known approximation guarantee for **PCSF** is 2.5415 obtained, as noted above, via Goemans' random-threshold idea applied to the threshold-rounding algorithm of Bienstock et al. This also shows that the integrality gap of (PCSF-LP) is at most 2.5415. The only known lower bound prior to this work was 2.

Our contributions

We demonstrate some limitations of (PCSF-LP) for designing approximation algorithms for **PCSF** and its special case, **PCST**, and in doing so dispel some widely-held beliefs about (PCSF-LP) and its specialization to **PCST**.

The integrality gap of (PCSF-LP) has been widely believed to be 2 since the work of Hajiaghayi and Jain [13], who devised a primal-dual 3-approximation algorithm for **PCSF** and pose the design of a primal-dual 2-approximation based on (PCSF-LP) as an open problem. However, as we show here, this belief is incorrect. Our main result is as follows.

▶ **Theorem 1.** *The integrality gap of* (PCSF-LP) *is at least* $9/4$*, even for planar instances of* **PCSF**. *Furthermore, any β-LMP approximation algorithm for the problem via* (PCSF-LP) *must have* $\beta \ge 4$.

When restricted to the non-prize-collecting Steiner forest problem, by setting $\pi_i = \infty$ for all i, (PCSF-LP) yields the standard LP for Steiner forest, which has an integrality gap of 2 [1]. Our result thus gives a clear separation between the integrality gaps of the prize-collecting and standard variants. It also shows a gap between the approximation ratios achievable relative to $\texttt{opt}_{\text{PCSF-LP}}$ by LMP and non-LMP approximation algorithms for **PCSF**. To the best of our knowledge, no such gaps were known previously for an LP for a natural network design problem. For example, for Steiner tree, there are no such gaps relative to the natural undirected LP obtained by specializing (PCSF-LP) to **PCST**. (There are however gaps in the current best approximation ratios known for Steiner tree and **PCST**, and approximation ratios achievable for **PCST** via LMP and non-LMP algorithms.)

In order to prove Theorem 1 we construct an instance on a large layered planar graph. Using a result of Carr and Vempala [6] it follows that (PCSF-LP) has a gap of α iff $\alpha \cdot (x, z)$ dominates a convex combination of integral solutions for any feasible solution (x, z). We show that this can only hold if $\alpha \ge 9/4$.

In his groundbreaking paper [15] introducing the iterative rounding method, Jain showed that extreme points x of the Steiner forest LP (and certain generalizations) have an edge e

with $x_e = 0$ or $x_e \geq 1/2$. This then immediately yields a 2-approximation algorithm for the underlying problem, by iteratively deleting an edge of value zero or rounding up an edge of value at least half to one and proceeding on the residual instance. Again, it was long believed that a similar structural result holds for **PCST**: extreme points of (PCST-LP) have an edge variable of value 0, or a variable of value at least 1/2. In fact, there were even stronger conjectures that envisioned the existence of a z-variable with value 1 in the case where all edge variables had positive value less than 1/2. We refute these conjectures.

▶ **Theorem 2.** *There exists an instance of* **PCST** *where (PCST-LP) has an extreme point with all edge variables positive and all variables having value at most* $1/3$*.*

In [14] it was shown, that for every vertex (x, z) of (PCSF-LP) (and hence also (PCST-LP)) where x is positive, there is at least one variable of value at least 1/3. Moreover for (PCSF-LP) this result is tight, i.e. there are instances of **PCSF** such that for some vertex (x, z) of (PCSF-LP), we have $x > \mathbb{0}$ and all coordinates are at most 1/3. However, no such example was known for (PCST-LP). We provide such an example for **PCST**, showing that the 1/3 upper bound on variable values is tight also for (PCST-LP).

2 The Integrality Gap for PCSF

2.1 Lower Bound on the Integrality Gap

We start proving Theorem 1 by describing the graph for our instance. Let P be a planar n-node 3-regular 3-edge-connected graph (for some large enough n to be determined later). Note that such graphs exist for arbitrarily large n; e.g., the graphs of simple 3-dimensional polytopes (such as planar duals of triangulations of a sphere) have these properties; they are 3-connected by Steinitz's theorem [19].

We obtain H from P by subdividing every edge e of P, so that e is replaced by a corresponding path with n internal nodes. Let r denote an arbitrary degree-3 node in H, and call it the root. Define $H^{(0)} := H$ and obtain $H^{(i)}$ from $H^{(i-1)}$ by attaching a copy of H to each degree-2 node v in $H^{(i-1)}$, identifying the root node of the copy with v; we call this the *copy of* H *with root* v. We also define the *parent* of any node $u \neq v$ in this copy to be v. In the end, we let $G := H^{(k)}$ for some large k, and we let r_0 be the node corresponding to the root of $H^{(0)}$. Figure 1 gives an example of this construction. Note that each copy of H can be thought of as a subgraph of G.

Next, let us define the source-sink pairs. We introduce a source-sink pair s, t whenever s and t are degree-3 nodes in the same copy of H. We also introduce a source-sink pair r_0, t whenever t is a degree-2 node in G.

Now let $x_e := 1/3$, for all $e \in E$, $z_{uv} := 0$ if u and v are degree-3 nodes in the same copy of H, and $z_{uv} := 1/3$ otherwise. (Here and henceforth, we abuse notation slightly and index z by the source-sink terminal pair that it corresponds to.) Clearly, (x, z) is a feasible solution for (PCSF-LP) by the 3-edge-connectivity of G.

Let α be the integrality gap of (PCSF-LP). By [6] there is a collection of forests $F_1, \ldots, F_q$ in G (the same forest could appear multiple times in the collection) such that picking a forest F uniformly at random from $F_1, \ldots, F_q$ satisfies

(a) $\mathbb{P}[e \in F] \leq \frac{\alpha}{3}$ for all $e \in E$, and

(b) Letting $u \sim_F v$ denote the event that u and v are connected in F, for all $u, v \in V(G)$, we have

$$\mathbb{P}[u \sim_F v] \geq (1 - \alpha z_{uv}) = \begin{cases} 1 & \text{if } u,\ v \text{ are degree-3 nodes in the same copy of } H, \\ 1 - \frac{\alpha}{3} & \text{if } u = r_0 \text{ and } v \text{ is a degree-2 node in } G. \end{cases}$$

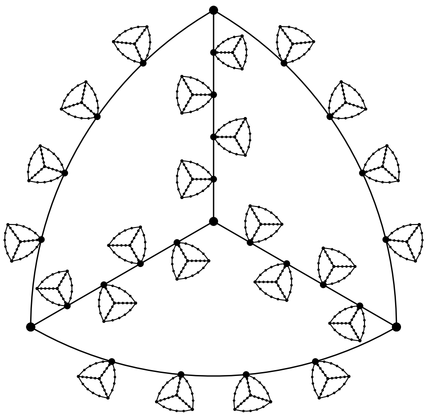


Figure 1 Taking $n = 4$, and hence P to be the complete graph on 4 vertices, the resulting graph $H^{(1)}$ is shown.

We begin by observing that we may assume that each forest $F_1, \ldots, F_q$ induces a tree when restricted to any of the copies of H in G. For consider any F_i, and a copy of H with root v; call this H'. Every degree-3 node in H' is connected to v in F_i, by requirement (b). So consider any degree-2 node u in H'. If u is not connected to a degree-3 node of H' (and hence to v) in F_i, then any edges of F_i adjacent to u can be safely deleted without destroying any connectivity amongst the source-sink pairs of the instance.

The argument will show that if α is too small, not all degree-2 nodes can be connected to r_0 with high enough probability. More precisely, we will show a geometrically decreasing probability, in k. The intuition is roughly as follows. Consider a copy H' of H with root u, where $u \neq r_0$. Almost all of the degree-2 nodes of H' that are connected to u in F will have degree 2 in F, since $F[H']$ is a tree and H' is made up of long paths. This is rather wasteful, since both edges adjacent to a typical degree-2 node v are used to connect; as each edge appears with probability $\alpha/3$, v can only be part of F (and hence connected to u) with probability about $\alpha/3$. Moreover, we will show that even conditioned on the event that u is *not* connected to r_0, there will be some choice of v such that v is connected to u in F with probability around $2/3$ (see (2) in Claim 3). This is again a waste in terms of connectivity to r_0. If p_i denotes the worst connectivity probability amongst nodes in $H^{(i)}$ in the construction, we have

$$p_{i+1} \lessapprox \tfrac{\alpha}{3} - \tfrac{2}{3}(1 - p_i). \qquad \text{((5) is a more precise version of this inequality)}$$

If $\alpha < 9/4$, this decreases geometrically, providing a counterexample for n large enough.

For now, let us introduce an abstract event I (that the reader may think of as "an ancestor of node v is not connected to r_0" motivated by the above discussion).

▶ **Claim 3.** *Let a forest F be picked uniformly at random from $F_1, \ldots, F_q$, let I be an event with $\mathbb{P}[I] > 0$ and let H' be a copy of H in G. Then there exists a degree-2 node v in H' such that*

$$\mathbb{P}\left[\deg_{F[H']}(v) = 1\right] \leq \frac{2}{n} \tag{1}$$

and

$$\mathbb{P}\left[Q_v \subseteq F \mid I\right] \geq \frac{2(n-1)}{3n}\,, \tag{2}$$

where Q_v is the path in H' corresponding to the edge of P containing v.

Proof. The event I corresponds to a nonempty multiset $\mathcal{F} \subseteq \{F_1, \ldots, F_q\}$ of the forests. Each of $F_1[H']$, $\ldots$, $F_q[H']$ is a tree, by our earlier assumption, and so each of them naturally induce a spanning tree of P. More precisely, for each $e \in E(P)$, let Q_e denote the corresponding path in H'; then $\{e \in E(P) : Q_e \subseteq F_i[H']\}$ is a spanning tree for each i. Thus

$$\sum_{e \in E(P)} |\{F' \in \mathcal{F} : Q_e \subseteq F'\}| = \sum_{F' \in \mathcal{F}} |\{e \in E(P) : Q_e \subseteq F'\}| = \sum_{F' \in \mathcal{F}} (n-1) = |\mathcal{F}|(n-1).$$

So there is an edge $f \in E(P)$ for which

$$\mathbb{P}\left[Q_f \subseteq F \,|\, I\right] = \frac{|\{F' \in \mathcal{F} : Q_f \subseteq F'\}|}{|\mathcal{F}|} \geq \frac{(n-1)}{|E(P)|} = \frac{2(n-1)}{3n}.$$

At most two of the nodes on Q_f are leaves in any of $F_1[H']$, $\ldots$, $F_q[H']$ (again since they are all trees). The total number of degree-2 nodes in H' lying on Q_f is n, so there exists a degree-2 node v in H' such that $v \in Q_f$ and $\mathbb{P}\left[\deg_{F[H']}(v) = 1\right] \leq \frac{2}{n}$. ◀

▶ **Claim 4.** *Let $\epsilon > 0$ be given. Then for n and k chosen sufficiently large, there exists a degree-2 node u in G such that*

$$\mathbb{P}\left[u \sim_F r_0\right] \leq \alpha - 2 + \epsilon\,,$$

where F is a uniformly random forest from $F_1, \ldots, F_q$.

Proof. Consider the root copy $H^{(0)}$ of H, with root r_0. Set $H_0 = H^{(0)}$. Pick a degree-2 node v in H_0 that satisfies (1) in Claim 3 for the trivial event $I := \{r_0 \sim_F r_0\}$ and $H' := H_0$. Let $r_1 := v$. Note that

$$\mathbb{P}\left[r_1 \sim_F r_0\right] = \mathbb{P}\left[\deg_{F[H_0]}(r_1) = 2\right] + \mathbb{P}\left[\deg_{F[H_0]}(r_1) = 1\right] \leq \frac{\alpha}{3} + \frac{2}{n} < 1\,.$$

The first inequality follows from (a) and (1), and the second since $\alpha \leq 2.5415$. Therefore $\mathbb{P}\left[r_1 \not\sim_F r_0\right] > 0$.

Suppose that we have defined $(H_0, r_1), (H_1, r_2), \ldots, (H_{i-1}, r_i)$ for some i with $1 \leq i \leq k$, such that the following hold for all $1 \leq j \leq i$: (i) r_j is a degree-2 node in H_{j-1}, and r_{j-1} is the root of H_{j-1}; (ii) $\mathbb{P}\left[r_{j-1} \not\sim_F r_0\right] > 0$ if $j \geq 2$; (iii) if $j \geq 2$, then (1) and (2) hold in Claim 3 for $H' = H_{j-1}$, $I = \{r_{j-1} \not\sim_F r_0\}$ and $v = r_j$. We now show how to define H_i and r_{i+1} such that the above properties continue to hold for $j = i + 1$.

First, set H_i to be the copy of H whose root is r_i. We have $\mathbb{P}\left[r_i \not\sim_F r_0\right] \geq \mathbb{P}\left[r_{i-1} \not\sim_F r_0\right] > 0$, so property (ii) continues to hold. Given this, pick a degree-2 node v in H_i that satisfies (1) and (2) in Claim 3 for the event $I := \{r_i \not\sim_F r_0\}$ and $H' := H_i$. Set $r_{i+1} := v$. Thus, properties (i) and (iii) continue to hold as well.

For $j \in \{0, \ldots, k\}$, due to the choice of r_{j+1} and (1), we have $\mathbb{P}\left[\deg_{F[H_j]}(r_{j+1}) = 1\right] \leq \frac{2}{n}$ and thus

$$\mathbb{P}\left[r_{j+1} \sim_F r_j\right] = \mathbb{P}\left[\deg_{F[H_j]}(r_{j+1}) = 2\right] + \mathbb{P}\left[\deg_{F[H_j]}(r_{j+1}) = 1\right] \leq \frac{\alpha}{3} + \frac{2}{n}\,. \tag{3}$$

For $j \in \{1, \ldots, k\}$, due to (2) and the choice of r_{j+1}, we get

$$\begin{aligned}\mathbb{P}\left[Q_{r_{j+1}} \subseteq F \,\wedge\, r_j \not\sim_F r_0\right] &= \mathbb{P}\left[Q_{r_{j+1}} \subseteq F \,|\, r_j \not\sim_F r_0\right] \cdot \mathbb{P}\left[r_j \not\sim_F r_0\right] \\ &\geq \frac{2(n-1)}{3n} \mathbb{P}\left[r_j \not\sim_F r_0\right] \\ &\geq \frac{2}{3}\big(1 - \mathbb{P}\left[r_j \sim_F r_0\right]\big) - \frac{2}{3n}\,. \end{aligned} \tag{4}$$

Hence, for $j \in \{0, \ldots, k\}$,

$$\begin{aligned}\mathbb{P}\left[r_{j+1} \sim_F r_0\right] &= \mathbb{P}\left[r_{j+1} \sim_F r_j \wedge r_j \sim_F r_0\right] \\ &= \mathbb{P}\left[r_{j+1} \sim_F r_j\right] - \mathbb{P}\left[r_{j+1} \sim_F r_j \wedge r_j \not\sim_F r_0\right] \\ &\leq \mathbb{P}\left[r_{j+1} \sim_F r_j\right] - \mathbb{P}\left[Q_{r_{j+1}} \subseteq F \wedge r_j \not\sim_F r_0\right] \\ &\leq \frac{\alpha}{3} + \frac{2}{n} - \frac{2}{3} + \frac{2}{3}\mathbb{P}\left[r_j \sim_F r_0\right] + \frac{2}{3n}, \qquad (5)\end{aligned}$$

where the first inequality follows from the fact that $r_{j+1} \sim_F r_j$ holds whenever $Q_{r_{j+1}} \subseteq F$ holds, and the second inequality follows from (3) and (4).

Expanding the recursion, we get

$$\mathbb{P}\left[r_{k+1} \sim_F r_0\right] \leq \left(\frac{\alpha-2}{3} + \frac{8}{3n}\right) \sum_{i=0}^{k} \left(\frac{2}{3}\right)^i + \left(\frac{2}{3}\right)^{k+1},$$

so for n and k large enough we obtain

$$\mathbb{P}\left[r_{k+1} \sim_F r_0\right] \leq \left(\frac{\alpha-2}{3} + \frac{\epsilon}{6}\right) \sum_{i=0}^{\infty} \left(\frac{2}{3}\right)^i + \frac{\epsilon}{2} = \alpha - 2 + \epsilon.$$

Since $u := r_{k+1}$ is a degree-2 node in G, the proof is complete. ◀

Now, we can prove the first part of Theorem 1. By Claim 4 and property (b) of the collection of forests, we get the inequality

$$\alpha - 2 \geq 1 - \alpha/3,$$

leading to $\alpha \geq 9/4$.

2.2 The Integrality Gap is Tight for the Construction

We note that for any n and k, the **PCSF** instance given by our construction has integrality gap at most 9/4. More generally, we show that the integrality gap over **PCSF** instances which admit a feasible solution (x, z) to (PCSF-LP) with $z_i \in \{0, 1/3\}$ for all i, is at most 9/4. (That is, the maximum ratio between the optimal values of the IP and the LP for such instances is at most 9/4.) This nicely complements our integrality-gap lower bound, and shows that our analysis above is tight (for such instances).

To show the first statement, we simply provide a distribution over forests $F_1, \ldots, F_q$ satisfying (a) and (b). (The next paragraph, which proves the second claim above, gives another proof.) Since $(2(n-1)/(3n)) \cdot \mathbb{1}$ is in the spanning tree polytope of P, there is a list of spanning trees such that every edge is contained in less than 2/3 of them. Consider the following distribution of forests. With probability $3 - \alpha$ we pick one of these spanning trees of P uniformly at random and subdivide it to obtain a tree in H; we take this tree in each copy of H to obtain a (non-spanning) tree in G. With probability $\alpha - 2$ we pick an arbitrary spanning tree of G. This random forest F satisfies

$$\mathbb{P}\left[e \in F\right] \leq (\alpha - 2) \cdot 1 + (3 - \alpha) \cdot \frac{2}{3} = \frac{\alpha}{3}.$$

Thus (a) holds for the above distribution. To see that (b) holds, note that for every degree-2 node v in G we have $\mathbb{P}\left[v \sim_F r_0\right] \geq \alpha - 2 = 1 - \alpha/3$.

For the second claim, we utilize threshold rounding to show that the integrality gap is at most 9/4 for such instances. Consider an instance of **PCSF** and a feasible point (x, z) for (PCSF-LP) such that the values of z-variables are 0 or γ for some fixed γ with $0 < \gamma < 1/2$. Using [1, 12], we can obtain an integer solution of cost at most $2c^\top x + \pi^\top z/\gamma$ by paying the penalties for all pairs with a non-zero z value. We can also obtain a solution of cost at most $2c^\top x/(1-\gamma)$ by connecting all pairs. Therefore, for any $p \in [0, 1]$, we can obtain an integer solution of cost at most

$$p\left(2c^\top x + \frac{\pi^\top z}{\gamma}\right) + (1-p)\left(\frac{2c^\top x}{1-\gamma}\right) \le \max\left\{\frac{2-2p\gamma}{1-\gamma}, \frac{p}{\gamma}\right\}(c^\top x + \pi^\top z)$$

showing that the integrality gap is at most

$$\mu := \min_{0\le p\le 1} \max\left\{\frac{2-2p\gamma}{1-\gamma}, \frac{p}{\gamma}\right\}.$$

The number μ is at most $2/(2\gamma^2 - \gamma + 1)$, which is equal to 9/4 for $\gamma = 1/3$. Note that for $\gamma = 1/4$ the $2/(2\gamma^2 - \gamma + 1)$ achieves its maximum value of 16/7.

2.3 Lagrangian-Multiplier Preserving Approximation Algorithms for PCSF

Recall that a β-Lagrangian-multiplier-preserving (LMP) approximation algorithm for **PCSF** is an approximation algorithm that returns a forest F satisfying

$$c(F) + \beta \cdot \pi(F) \le \beta \cdot \texttt{opt}\,.$$

We show that we must have $\beta \ge 4$ in order to obtain a β-LMP algorithm relative to the optimum of the LP-relaxation (PCSF-LP), that is, to obtain the guarantee $c(F) + \beta \cdot \pi(F) \le \beta \cdot \texttt{opt}_{\text{PCSF-LP}}$. To obtain this lower bound, we modify our earlier construction slightly. We construct $G = H^{(k)}$ in a similar fashion as before, but we now choose P (the "base graph") to be an n-node l-regular l-edge-connected graph. Let $x_e := 1/l$ for all $e \in E$, and let $z_{uv} := 0$ if u and v are degree-l nodes in the same copy of H, and $z_{uv} := 1 - 2/l$ otherwise.

By arguments similar to [6] (see, e.g., the proof of Theorem 7.2 in [10], and Theorem 8 in the Appendix), one can show that if there exists a β-LMP approximation algorithm for **PCSF** relative to (PCSF-LP) then there are forests F_1,..., F_q in G (the same forest could appear multiple times) such that picking a forest F uniformly at random from $F_1, \dots, F_q$ satisfies

(a') $\mathbb{P}[e \in F] \le \frac{\beta}{l}$ for all $e \in E$, and

(b') $\mathbb{P}[u \sim_F v] \ge (1 - z_{uv}) = \begin{cases} 1 & u,\ v \text{ are degree-}l\text{ nodes in the same copy of } H \\ \frac{2}{l} & \text{if } u = r_0 \text{ and } v \text{ is a degree-2 node in } G \end{cases}$

for all $u, v \in V(G)$.

It is straightforward to obtain the analogues of Claim 3 and Claim 4.

► **Claim 5.** *Let a forest F be picked uniformly at random from $F_1, \dots, F_q$, let I be an event with $\mathbb{P}[I] > 0$ and let H' be a copy of H in G. There exists a degree-2 node v in H', such that*

$$\mathbb{P}\left[\deg_{F[H']}(v) = 1\right] \le \frac{2}{n} \tag{6}$$

and

$$\mathbb{P}[Q_v \subseteq F \mid I] \ge \frac{2(n-1)}{ln}, \tag{7}$$

where Q_v is the path in H' that contains v and corresponds to an edge of P.

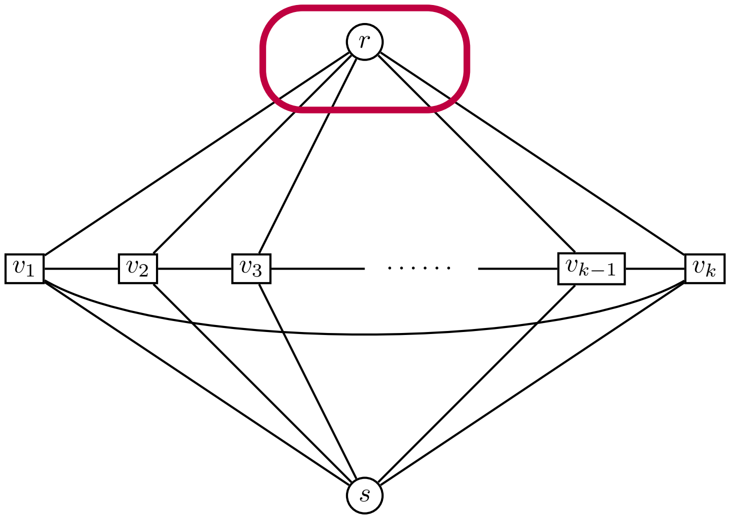


Figure 2 Here, each of the nodes $v_1,\ldots,\ v_k$ corresponds to the gadget in Figure 3. Additionally, a cut $\{r\}$ is marked as a tight constraint in (PCST-LP) for the constructed point (x, z). $x_e = 1/k$ for all edges e.

▶ **Claim 6.** *Let $\epsilon > 0$ be given. Then for n and k sufficiently large, and choosing F uniformly at random from $F_1, \ldots, F_q$, there exists a degree-2 node u in G such that*

$$\mathbb{P}\left[u \sim_F r_0\right] \leq \frac{\beta - 2}{l - 2} + \epsilon\,.$$

For the node u from Claim 6, we have $(\beta - 2)/(l - 2) \geq \mathbb{P}\left[v \sim_F r_0\right] \geq 2/l$. Thus, β is at least $4 - 4/l$, which approaches 4 as l increases. This completes the proof of the second part of Theorem 1.

Moreover, the analysis is tight for the above construction. For a solution to (PCSF-LP) where z takes on only two distinct values, say 0 and γ, threshold rounding shows that for $\beta = 2 + 2\gamma < 4$ the desired collection of forests exists. However, for an unbounded number of distinct values of z, no constant-factor upper bound is known.

3 An Extreme Point for PCST with All Values at most $\frac{1}{3}$

In this section we present a proof of Theorem 2. Take an integer $k \geq 4$ and consider the graph $G = (V, E)$ in Figure 2. Here, the nodes $v_1,\ldots,\ v_k$ represent the gadgets shown in Figure 3. The gadget consists of ten nodes, and there are precisely four edges incident to a node in the gadget. We let r to be the root node and introduce a source-sink node pair (v, r) for every node $v \in V \setminus \{r\}$.

In the case $k = 6$, the next claim proves Theorem 2.

▶ **Claim 7.** *The following is an extreme point of (PCST-LP) for this instance: $z_s = 0$ and $z_u = 1 - 4/k$ for every node u in $V \setminus \{r, s\}$. For the wavy edges in Figure 3, we have $x_{u_1u_2} := x_{u_3u_4} := x_{u_5u_6} := x_{u_7u_8} := x_{u_9u_{10}} := 2/k$, and $x_e = 1/k$ for all the other edges e.*

Proof. It is straightforward to check that the defined point (x, z) is feasible. Let us show that the defined point (x, z) is a vertex of (PCST-LP). To show this, it is enough to provide a set of tight constraints in (PCST-LP) which uniquely define the above point (x, z).

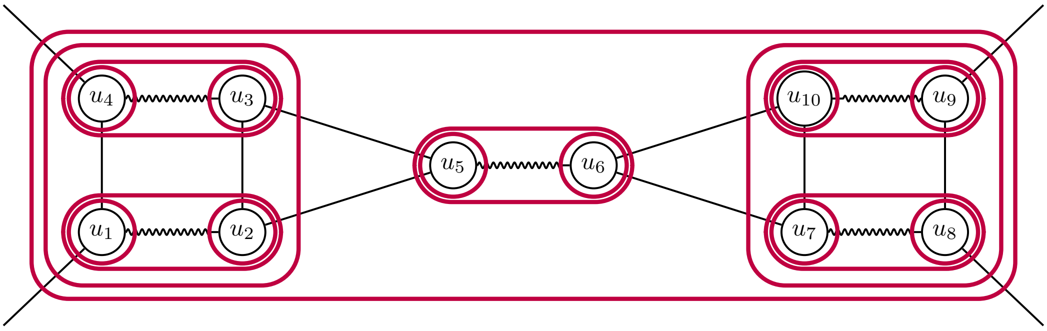


Figure 3 A gadget used for the construction in Figure 2. Additionally, the cuts are marked as tight constraints in (PCST-LP) for the constructed point (x, z). For an edge e, $x_e = 2/k$ if e is a wavy edge, and $x_e = 1/k$ if it is a straight edge.

Let us consider the gadget in Figure 3. For each such gadget, the set of tight inequalities from (PCST-LP) contains the following constraints:

$$x(\delta(u_i)) + z_{u_i} = 1 \qquad \forall i \in \{1, \dots, 10\} \tag{8}$$

$$x(\delta(\{u_1, \dots, u_{10}\})) + z_{u_i} = 1 \qquad \forall i \in \{1, \dots, 10\} \tag{9}$$

$$x(\delta(\{u_i, u_{i+1}\})) + z_{u_i} = 1 \qquad \forall i \in \{1, 3, 5, 7, 9\} \tag{10}$$

$$x(\delta(\{u_1, \dots, u_4\})) + z_{u_1} = 1 \tag{11}$$

$$x(\delta(\{u_7, \dots, u_{10}\})) + z_{u_7} = 1\,. \tag{12}$$

There are two more tight constraints which we use in the proof:

$$x(\delta(r)) + z_s = 1 \tag{13}$$

$$z_s = 0\,. \tag{14}$$

Let us prove that the constraints (8)–(14) define the point (x, z) from Claim 7. First, let us consider a gadget in Figure 3. It is clear that (9) implies $z_{u_1} = \ldots = z_{u_{10}}$. By (8) and (10), we get

$$2x_{u_1u_2} = x(\delta(u_1)) + x(\delta(u_2)) - x(\delta(\{u_1, u_2\})) = (1 - z_{u_1}) + (1 - z_{u_1}) - (1 - z_{u_1}) = (1 - z_{u_1})\,,$$

and hence $x_{u_1u_2} = (1 - z_{u_1})/2$. Similarly, we obtain $x_{u_1u_2} = x_{u_3u_4} = \ldots = x_{u_9u_{10}} = (1 - z_{u_1})/2$.

Now, we have

$$x_{u_3u_5} + x_{u_2u_3} = x(\delta(u_3)) - x_{u_3u_4} = (1 - z_{u_1})/2$$
$$x_{u_2u_3} + x_{u_2u_5} = x(\delta(u_2)) - x_{u_1u_2} = (1 - z_{u_1})/2$$
$$x_{u_2u_5} + x_{u_3u_5} = x(\delta(u_5)) - x_{u_5u_6} = (1 - z_{u_1})/2\,,$$

implying $x_{u_2u_3} = x_{u_2u_5} = x_{u_3u_5} = (1 - z_{u_1})/4$. Similarly, $x_{u_6u_7} = x_{u_6u_{10}} = x_{u_7u_{10}} = (1 - z_{u_1})/4$.

By (10) and (11), we get

$$2x_{u_1u_4} = x(\delta(\{u_1, u_2\})) + x(\delta(\{u_3, u_4\})) - x(\delta(\{u_1, \ldots, u_4\})) - 2x_{u_2u_3} = (1 - z_{u_1})/2\,,$$

showing $x_{u_1u_4} = (1 - z_{u_1})/4$. Similarly, we get $x_{u_8u_9} = (1 - z_{u_1})/4$. From here, it is straightforward to show that all straight edges in Figure 3 have value $(1 - z_{u_1})/4$ and all wavy edges have value $(1 - z_{u_1})/2$.

Consider the graph in Figure 2. Due to the edge v_1v_2, the straight edges in the gadget associated to v_1 have the same x value as the straight edges in the gadget associated to v_2. Thus, due to the cycle $v_1v_2\dots v_k$ the straight edges in all gadgets have the same x value. To finish the proof use (13) and (14). ◀

Acknowledgement. We would like to thank Hausdorff Research Institute for Mathematics. This research was initiated during the Hausdorff Trimester Program "Combinatorial Optimization".

References

1 A. Agrawal, P. Klein, and R. Ravi. When trees collide: an approximation algorithm for the generalized Steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456, 1995.

2 A. Archer, M. Bateni, M. Hajiaghayi, and H. Karloff. Improved approximation algorithms for prize-collecting Steiner tree and TSP. *SIAM Journal on Computing*, 40(2):309–332, 2011.

3 D. Bienstock, M. X. Goemans, D. Simchi-Levi, and D. Williamson. A note on the prize collecting traveling salesman problem. *Mathematical Programming*, 59(1):413–420, 1993.

4 A. Blum, R. Ravi, and S. Vempala. A constant-factor approximation algorithm for the k-MST problem. *Journal of Computer and System Sciences*, 58(1):101–108, 1999.

5 J. Byrka, F. Grandoni, T. Rothvoss, and L. Sanità. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM*, 60(1):6, 2013.

6 R. D. Carr and S. Vempala. On the Held-Karp relaxation for the asymmetric and symmetric traveling salesman problems. *Mathematical Programming A*, 100:569–587, 2004.

7 M. Chlebík and J. Chlebíková. The Steiner tree problem on graphs: Inapproximability results. *Theoretical Computer Science*, 406(3):207–214, 2008. `doi:10.1016/j.tcs.2008.06.046`.

8 F. A. Chudak, T. Roughgarden, and D. P. Williamson. Approximate k-MSTs and k-Steiner trees via the primal-dual method and Lagrangean relaxation. *Mathematical Programming*, 100(2):411–421, 2004.

9 N. Garg. Saving an epsilon: a 2-approximation algorithm for the k-MST problem in graphs. In *Proceedings of the 37th ACM Symposium on Theory of Computing*, pages 396–402, 2005.

10 K. Georgiou and C. Swamy. Black-box reductions for cost-sharing mechanism design. *Games and Economic Behavior*, 2013. `doi:10.1016/j.geb.2013.08.012`.

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 ACM Symposium on Theory of Computing*, pages 1161–1176, 2012.

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

13 M. Hajiaghayi and K. Jain. The prize-collecting generalized Steiner tree problem via a new approach of primal-dual schema. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 631–640, 2006.

14 M. Hajiaghayi and A. Nasri. Prize-collecting Steiner networks via iterative rounding. In *Theoretical Informatics: LATIN 2010*, pages 515–526. Springer, 2010.

15 K. Jain. A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica*, 21(1):39–60, 2001. `doi:10.1007/s004930170004`.

16 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.

17 R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Springer, 1972.

18 J. Könemann, O. Parekh, and D. Segev. A unified approach to approximating partial covering problems. *Algorithmica*, 59(4):489–509, 2011.

19 E. Steinitz. Polyeder und Raumeinteilungen. In *Enzyclopädie der Mathematischen Wissenschaften, vol. 3, Geometrie, erster Teil, zweite Hälfte*, pages 1–139. Teubner, 1922.

20 D. P. Williamson and D. B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2010.

A Implications of an LMP Approximation Algorithm for PCSF

We adapt the arguments in [6] to show that a β-LMP approximation relative to (PCSF-LP) implies that any fractional solution (x, z) to (PCSF-LP) can be translated to a distribution over integral solutions to (PCSF-LP) satisfying certain properties; this implies the existence of the forests $F_1, \dots, F_q$ in Section 2.3. The arguments below are known (see, e.g., the proof of Theorem 7.2 in [10]); we include them for completeness.

Let $G = (V, E)$, $\{c_e \geq 0\}_{e \in E}$, $\{(s_i, t_i, \pi_i)\}_{i=1}^k$ be a **PCSF**-instance. Let $\{(x^{(q)}, z^{(q)})\}_{q \in \mathcal{I}}$ be the set of all integral solutions to (PCSF-LP), where $\mathcal{I}$ is simply an index set.

▶ **Theorem 8.** *Let $\mathcal{A}$ be a β-LMP approximation algorithm for* **PCSF** *relative to* (PCSF-LP)*. Given any fractional solution (x^*, z^*) to* (PCSF-LP)*, one can obtain nonnegative multipliers $\{\lambda^{(q)}\}_{q \in \mathcal{I}}$ such that $\sum_q \lambda^{(q)} = 1$, $\sum_q \lambda^{(q)} x^{(q)} \leq \beta x^*$, and $\sum_q \lambda^{(q)} z^{(q)} \leq z^*$. Moreover, the $\lambda^{(q)}$ values are rational if (x^*, z^*) is rational.*

Proof. Consider the following pair of primal and dual LPs.

$$\begin{aligned} \max \quad & \sum_q \lambda^{(q)} & & \text{(P)} \\ \text{s.t.} \quad & \sum_q \lambda^{(q)} x_e^{(q)} \leq \beta x_e^* & \forall e & \\ & \sum_q \lambda^{(q)} z_i^{(q)} \leq z_i^* & \forall i & \\ & \sum_q \lambda^{(q)} \leq 1 & & \\ & \lambda \geq 0. & & \end{aligned}$$

$$\begin{aligned} \min \quad & \sum_e \beta x_e^* d_e + \sum_i z_i^* \rho_i + \gamma & & \text{(D)} \\ \text{s.t.} \quad & \sum_e x_e^{(q)} d_e + \sum_i z_i^{(q)} \rho_i + \gamma \geq 1 & \forall q & \\ & d, \rho, \gamma \geq 0. & & \end{aligned}$$

It suffices to show that the optimal value of (P) is 1. The rationality of the $\lambda^{(q)}$ values when (x^*, z^*) is rational then follows from the fact that an LP with rational data has a rational optimal solution. (The proof below also yields a polynomial-time algorithm to solve (P) by showing that $\mathcal{A}$ can be used to obtain a separation oracle for the dual.)

Note that both (P) and (D) are feasible, so they have a common optimal value. We show that $\texttt{opt}_D = 1$. Setting $\gamma = 1$, $d = \rho = \mathbb{0}$, we have that $\texttt{opt}_D \leq 1$. Suppose (d, ρ, γ) is feasible to (D) and $\sum_e \beta x_e^* d_e + \sum_i z_i^* \rho_i + \gamma < 1$. Consider the **PCSF** instance given by G, edge costs $\{d_e\}_{e \in E}$, and terminal pairs and penalties $\{(s_i, t_i, \rho_i/\beta)\}_{i=1}^k$. Running $\mathcal{A}$ on this instance, we can obtain an integral solution $(x^{(q)}, z^{(q)})$ such that

$$\sum_e d_e x_e^{(q)} + \sum_i \rho_i z_i^{(q)} + \gamma \leq \beta \Big(\sum_e d_e x_e^* + \sum_i z_i^* \rho_i / \beta \Big) + \gamma < 1$$

which contradicts the feasibility of (d, ρ, γ). Hence, $\texttt{opt}_D = 1$. ◀

Note that if (x^*, z^*) is rational, then since the $\lambda^{(q)}$ values are rational, we can multiply them by a suitably large number to convert them to integers; thus, we may view the distribution specified by the $\lambda^{(q)}$ values as the *uniform distribution* over a *multiset* of integral solutions to (PCSF-LP).

We remark that the converse of Theorem 8 also holds in the following sense. If for every fractional solution (x^*, z^*) to (PCSF-LP), we can obtain $\lambda^{(q)}$ values (or equivalently, a distribution over integral solutions to (PCSF-LP)) satisfying the properties in Theorem 8, then we can obtain a β-LMP approximation algorithm for **PCSF** relative to (PCSF-LP): this follows, by simply returning the integral solution $(x^{(q)}, z^{(q)})$ with $\lambda^{(q)} > 0$ that minimizes $\sum_e c_e x_e^{(q)} + \beta \sum_i \pi_i z_i^{(q)}$.

Approximating Unique Games Using Low Diameter Graph Decomposition

Vedat Levi Alev[*1] **and Lap Chi Lau**[†2]

1 University of Waterloo, Waterloo, ON, Canada
vlalev@uwaterloo.ca

2 University of Waterloo, Waterloo, ON, Canada
lapchi@uwaterloo.ca

Abstract

We design approximation algorithms for Unique Games when the constraint graph admits good low diameter graph decomposition. For the $\mathsf{Max\text{-}2Lin}_k$ problem in K_r-minor free graphs, when there is an assignment satisfying $1-\varepsilon$ fraction of constraints, we present an algorithm that produces an assignment satisfying $1-O(r\varepsilon)$ fraction of constraints, with the approximation ratio independent of the alphabet size. A corollary is an improved approximation algorithm for the $\mathsf{Min\text{-}UnCut}$ problem for K_r-minor free graphs. For general Unique Games in K_r-minor free graphs, we provide another algorithm that produces an assignment satisfying $1-O(r\sqrt{\varepsilon})$ fraction of constraints.

Our approach is to round a linear programming relaxation to find a minimum subset of edges that intersects all the inconsistent cycles. We show that it is possible to apply the low diameter graph decomposition technique on the constraint graph directly, rather than to work on the label extended graph as in previous algorithms for Unique Games. The same approach applies when the constraint graph is of genus g, and we get similar results with r replaced by $\log g$ in the $\mathsf{Max\text{-}2Lin}_k$ problem and by $\sqrt{\log g}$ in the general problem. The former result generalizes the result of Gupta-Talwar for Unique Games in the $\mathsf{Max\text{-}2Lin}_k$ case, and the latter result generalizes the result of Trevisan for general Unique Games.

1998 ACM Subject Classification G.2.2 Graph Theory

Keywords and phrases Unique Games, Low Diameter Graph Decomposition, Bounded Genus Graphs, Fixed Minor Free Graphs, Approximation Algorithms, Linear Programming

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.18

1 Introduction

For a given integer $k \geq 1$, an undirected graph $G=(V,E)$ and a set $\Pi = \{\pi_{uv} : uv \in E\}$ of permutations on $[k]$ satisfying $\pi_{uv} = \pi_{vu}^{-1}$, the Unique Games problem with alphabet size k (denoted by UG_k) is the problem of finding an assignment $x : V \to [k]$ to the vertices such that the number of edges $e = uv \in E$ satisfying the constraint $\pi_{uv}(x(u)) = x(v)$ is maximized. The value $\mathsf{SAT}(\mathfrak{I})$ of a Unique Games instance $\mathfrak{I} = (G, \Pi)$ is defined as,

$$\mathsf{SAT}(\mathfrak{I}) = \max_{x:V\to[k]} \frac{1}{|E|} \sum_{uv\in E} \mathbf{1}[\pi_{uv}(x(u)) = x(v)]$$

[*] Supported by the GO-Bell Scholarship and the David R. Cheriton Graduate Scholarship.
[†] Supported by NSERC Discovery Grant 2950-120715 and NSERC Accelerator Supplement 2950-120719.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 18; pp. 18:1–18:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

i.e. the maximum fraction of satisfiable constraints over all assignments x. We define $\mathsf{UNSAT}(\mathfrak{I}) = 1 - \mathsf{SAT}(\mathfrak{I})$ as the minimum fraction of unsatisfied constraints.

The Unique Games Conjecture of Khot [23] postulates that it is **NP**-hard to distinguish whether a given instance $\mathfrak{I} = (G, \Pi)$ of the Unique Games problem is almost satisfiable or almost unsatisfiable, and the problem becomes harder as the alphabet size k increases.

▶ **Conjecture 1** (The Unique Games Conjecture, [23]). *For every $\varepsilon > 0$, there exists an integer $k := k(\varepsilon)$, such that the decision problem of whether an instance $\mathfrak{I}$ of UG_k satisfies $\mathsf{SAT}(\mathfrak{I}) \geq 1 - \varepsilon$ or $\mathsf{SAT}(\mathfrak{I}) \leq \varepsilon$ is **NP**-hard.*

The Unique Games Conjecture has attracted much attention over the years, due to its implications regarding the hardness of approximation for many **NP**-hard problems [25, 24, 31]. An important case of Unique Games is the $\mathsf{Max\text{-}2Lin}_k$ problem when the constraints are of the form $x_u - x_v \equiv c_{uv} \pmod{k}$ for $uv \in E$. This problem is shown to be as hard as the general case of the Unique Games problem by Khot et al. [24]. The Max-Cut problem is a well-studied special case of $\mathsf{Max\text{-}2Lin}_2$ where $x_u - x_v \equiv 1 \pmod{2}$ for $uv \in E$. Assuming the Unique Games Conjecture, Khot et al. [24] proved that it is **NP**-hard to distinguish Max-Cut instances where the optimal value is at least $1 - \varepsilon$ from instances where the optimal value is at most $1 - \Theta(\sqrt{\varepsilon})$.

There have been several efforts in designing polynomial time approximation algorithms for Unique Games [23, 36, 18, 11, 12], where the objective is to minimize the number of unsatisfied constraints. Let $\mathfrak{I}$ be the given instance of UG_k with n variables and $\mathsf{UNSAT}(\mathfrak{I}) = \varepsilon$. Trevisan [36] gave an SDP-based algorithm that provides an assignment which violates at most an $\mathcal{O}(\sqrt{\varepsilon \log n})$ fraction of the constraints. Gupta and Talwar [18] gave an LP-based algorithm that provides an assignment which violates at most an $\mathcal{O}(\varepsilon \cdot \log n)$ fraction of the constraints. Charikar, Makarychev, and Makarychev [11] gave an SDP-based algorithm which finds an assignment violating at most a $\mathcal{O}(\sqrt{\varepsilon \log k})$ fraction of constraints, where k is the alphabet size. Chlamtac, Makarychev, and Makarychev [12] gave another SDP-based algorithm which finds an assignment violating at most an $\mathcal{O}(\varepsilon \cdot \sqrt{\log k \log n})$-fraction of the constraints.

There are also some previous works exploiting the structures of the constraint graphs. Arora, Barak and Steurer [5] presented a subexponential time algorithm to distinguish the two cases in the Unique Games Conjecture. Their approach uses the spectral information of the constraint graph. If the Laplacian matrix of the constraint graph has only a few small eigenvalues, then they extend the subspace enumeration approach of Kolla [27] to search over this eigenspace for a good assignment. On the other hand, if there are many small eigenvalues, they give a graph decomposition procedure to delete a small fraction of edges so that each component in the remaining graph has only a few small eigenvalues. Combining these two steps carefully gives their subexponential time algorithm. There is also an SDP-based propagation rounding approach to find a good assignment when the constraint graph is an expander [6] and more generally when the Laplacian matrix of the constraint graph has only a few small eigenvalues [9, 19]. These gave an alternative SDP-based subexponential time algorithm for the Unique Games Conjecture.

Our initial motivation is to study the Unique Games problem when the Laplacian matrix of the constraint graph has many small eigenvalues, as there are no known good approximation algorithms for Unique Games in these graphs. The most natural graph family possessing this property is the class of graphs without a K_r minor, where a graph H is a minor of G if H can be obtained from G by deleting and contracting edges, and K_r is the complete graph with r vertices. Kelner et al. [22], after a sequence of works [10, 34, 21], proved that the k-th smallest eigenvalue of the Laplacian matrix of a bounded degree K_r-minor free graph is $\mathcal{O}(\text{poly}(r) \cdot k/n)$, showing that there are many small eigenvalues. The class of K_r-minor free graphs is well studied and is known to contain the class of planar graphs and the class of bounded genus graphs, where a graph is of genus g if the graph can be embedded into a surface having at most g handles without edge crossings. There are different (non-spectral) techniques in designing approximation algorithms for various problems in K_r-minor free graphs (see e.g. [14, 13]), including problems that are known to be harder than Unique Games. This leads us to the question of whether we can extract those ideas to design better algorithms for Unique Games.

1.1 Our Results

In this paper, we consider the problem of approximately minimizing the number of unsatisfied constraints in an UG_k instance $\mathfrak{I} = (G, \Pi)$, when the constraint graph G is K_r-minor free. Our first theorem is for the $\mathsf{Max\text{-}2Lin}_k$ problem.

▶ **Theorem 2.** *Given a $\mathsf{Max\text{-}2Lin}_k$ instance $\mathfrak{I} = (G, \Pi)$ where G is a K_r-minor free graph and $\mathsf{UNSAT}(\mathfrak{I}) = \varepsilon$ (respectively where G is of genus at most g), there is an LP-based polynomial time algorithm which outputs an assignment that violates at most an $\mathcal{O}(r \cdot \varepsilon)$ fraction of constraints (respectively at most a $\mathcal{O}(\log g \cdot \varepsilon)$ fraction of constraints).*

For $\mathsf{Max\text{-}2Lin}$, Theorem 2 on bounded genus graphs is a refinement of the $\mathcal{O}(\log n \cdot \varepsilon)$ bound of Gupta and Talwar [18] as $g = \mathcal{O}(n)$. Theorem 2 also implies an improved approximation algorithm for the $\mathsf{Min\text{-}Uncut}$ problem (the complement of the $\mathsf{Max\text{-}Cut}$ problem), where the objective is to delete a minimum subset of edges so that the resulting graph is bipartite.

▶ **Corollary 3.** *There is an LP-based polynomial time $O(r)$-approximation algorithm (respectively a $O(\log g)$-approximation algorithm) for the $\mathsf{Min\text{-}Uncut}$ problem for K_r-minor free graphs (respectively for graphs of genus g).*

The best known approximation algorithm for $\mathsf{Min\text{-}Uncut}$ is an SDP-based $\mathcal{O}(\sqrt{\log n})$-approximation algorithm [2, 12]. We are not aware of any improvement of this bound for K_r-minor free graphs and bounded genus graphs. The above algorithms crucially used the symmetry of the linear constraints in $\mathsf{Max\text{-}2Lin}$. For general Unique Games, we present a different algorithm with weaker guarantees. The following theorem on bounded genus graphs is a refinement of the $\mathcal{O}(\sqrt{\varepsilon \cdot \log n})$ bound of Trevisan [36] (see the discussion in [18, Section 4]).

▶ **Theorem 4.** *Given a UG_k instance $\mathfrak{I} = (G, \Pi)$ where G is a K_r-minor free graph and $\mathsf{UNSAT}(\mathfrak{I}) = \varepsilon$ (respectively where G is of genus at most g), there is an LP-based polynomial time algorithm which outputs an assignment that violates at most an $\mathcal{O}(r \cdot \sqrt{\varepsilon})$ fraction of constraints (respectively at most a $\mathcal{O}(\sqrt{\log g \cdot \varepsilon})$ fraction of constraints).*

The main tool in our algorithms is the low diameter graph decomposition for K_r-minor free graphs and bounded genus graphs (see Section 2). Both of our algorithms are LP-based. The $\mathsf{Max\text{-}2Lin}_k$ algorithm is based on cutting inconsistent cycles, which is different from most existing algorithms for Unique Games that are based on finding good assignments. The UG_k algorithm is based on the propagation rounding method in Gupta and Talwar [18]. We defer the technical overviews to Section 3.2 and Section 5.2, after the preliminaries are defined.

1.2 Related Work

There are polynomial time approximation schemes for many problems in K_r-minor free graphs (see [13, 14]). For example, there is a $(1 - \varepsilon)$-approximation algorithm for $\mathsf{Max\text{-}Cut}$

with running time $2^{1/\varepsilon} \cdot n^{\mathcal{O}_r(1)}$ for K_r-minor free graphs. The approach is a generalization of Baker's approach for planar graphs [7] remaining graph is of bounded treewidth, and then using dynamic programming to solve the problem on each bounded treewidth component. This approach can be used to distinguish the two cases in the Unique Games Conjecture for K_r-minor free graphs for any fixed r. However, this approach is not applicable to obtain multiplicative approximation algorithms for minimizing the number of unsatisfied constraints for Unique Games, since it requires to remove a constant fraction of edges while the optimal value could be very small. As mentioned previously, we are not aware of any polynomial time approximation algorithms with performance ratio better than $\mathcal{O}(\sqrt{\log n})$ for the Min-Uncut problem for K_r-minor free graphs.

The low diameter graph decomposition technique is very useful in designing approximation algorithms for K_r-minor free graphs. It was first developed by Klein, Plotkin and Rao [26] to establish the multicommodity flow-cut gap of K_r-minor free graphs, and since then this technique has found numerous applications. A recent result using this technique is a $(\mathcal{O}_\varepsilon(r^2), 1+\varepsilon)$ bicriteria approximation algorithm [8] for the small set expansion problem, which is shown to be closely related to the Unique Games problem [32, 33].

It is a well-known result of Hadlock [20] that the maximum cut problem can be solved exactly in polynomial time on planar graphs. In Agarwal's thesis [3], he showed that an SDP relaxation (with triangle inequalities) for UG_2 is exact for planar graphs, using a multicommodity flow-cut type argument introduced in Agarwal et al. [4]. It is mentioned in [3] that this approach of bounding the integrality gap (even approximately) is only known to work for K_5-minor free graphs.

Steurer and Vishnoi [35] showed that the Unique Games problem can be reduced to the Multicut problem and used it to recover Gupta and Talwar's result in the case of Max-2Lin$_k$. The approach of Steurer and Vishnoi is similar to ours; see Section 3.2 for some discussion.

1.3 Organization

In Section 2, we describe the low diameter graph decomposition results that we will apply. In Section 3, we first present the proof for the Min-Uncut problem, as it is simpler and illustrates all the main ideas. Then we generalize the proof to the Max-2Lin$_k$ problem in Section 4. In Section 5, we show the result for general Unique Games. The proof overviews for Theorem 2 and Theorem 4 will be presented in the corresponding sections, Section 3.2 and Section 5.2, after the preliminaries are defined.

2 Low Diameter Graph Decompositions

Let $G = (V, E)$ be a graph with non-negative edge weights $w : E \to \mathbb{R}_+$. A collection $P = \{C_1, \ldots, C_k\}$ of disjoint subsets $C_j \subseteq V$ (called clusters) is a partition if they satisfy $V = \cup_{i=1}^k C_j$. We call a partition P weakly Δ-bounded if each of the clusters has weak diameter Δ, i.e.

$$d_G(u, v) \leq \Delta \quad \forall u, v \in C_j; \forall j \in [k]$$

where d_G denotes the shortest path distance on G (induced by the edge weights w). We say that the partition P is strongly Δ-bounded if each cluster has strong diameter Δ, i.e.

$$d_{G[C_j]}(u, v) \leq \Delta \quad \forall u, v \in C_j; \forall j \in [k]$$

where $d_{G[C_j]}$ denotes the shortest path distance in the induced subgraph $G[C_j]$.

We write $P(u)$ for the unique cluster C_j containing the vertex $u \in V$. We call a distribution $\mathcal{A}$ of partitions Δ-bounded D-separating if each cluster is of diameter Δ and for each edge $uv \in E$ we have

$$\mathbb{P}_{P\sim\mathcal{A}}[P(u) \neq P(v)] \leq \frac{D}{\Delta} \cdot w(u,v). \tag{1}$$

This implies that we can cut a graph into clusters with diameter at most Δ by deleting all the inter-cluster edges, while only losing a D/Δ fraction of the total edge weight.

We call a Δ-bounded D-separating partitioning scheme efficient, if we can sample it in polynomial time.

The seminal work of Klein, Plotkin and Rao [26] showed the first low diameter graph decomposition scheme for planar graphs and more generally for K_r-minor free graphs. We will use the latest result of this line of work [26, 16, 29, 1], as it gives the best known quantitative bound and also it guarantees the clusters have strong diameter Δ which will be important in our algorithm for general Unique Games.

▶ **Theorem 5** ([1]). *Every weighted K_r-minor free graph admits an efficient weakly Δ-bounded $\mathcal{O}(r)$-separating partitioning scheme for any $\Delta \geq 0$.*

▶ **Theorem 6** ([1]). *Every weighted K_r-minor free graph admits an efficient strongly Δ-bounded $\mathcal{O}(r^2)$-separating partitioning scheme for any $\Delta \geq 0$.*

We will also use the optimal bounds for bounded genus graphs, to derive better results for Unique Games in these graphs.

▶ **Theorem 7** ([1, 29]). *Every weighted graph of genus g admits an efficient strongly Δ-bounded $\mathcal{O}(\log g)$-separating partitioning scheme for any $\Delta \geq 0$.*

The results in [1] are stated using the language of padded decompositions, but it is easy to see that the results we stated are corollaries of the theorems in [1].

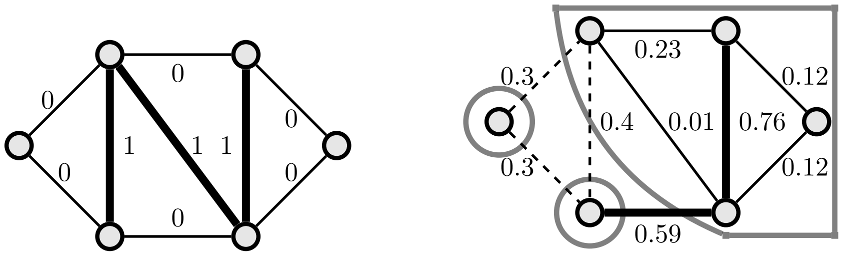
3 Minimum Uncut

Given an undirected graph $G = (V, E)$ with a non-negative cost c_e for each edge $e \in E$, the Min-Uncut problem is to find a subset $S \subseteq V$ to minimize the total cost of the uncut edges (the edges with both endpoints in S or both endpoints in $V - S$). Alternatively, the problem is equivalent to finding a subset $F \subseteq E$ of minimum total cost so that $G - F$ is a bipartite graph (so F is the uncut edges). As a graph is bipartite if and only if it has no odd cycles, the problem is equivalent to finding a subset of edges of minimum total cost that intersects all the odd cycles in the graph, which is also known as the Odd Cycle Transversal problem. We will tackle the Min-Uncut problem using this perspective, by writing a linear program for the Odd Cycle Transversal problem.

As mentioned already, the Min-Uncut problem is a special case of Max-2Lin$_2$. We will see in Section 4 that the ideas in this section can be readily generalized to design an approximation algorithm for the Max-2Lin$_k$ problem.

3.1 Linear Programming Relaxation

We consider the following well-known linear programming relaxation for the Odd Cycle Transversal problem, which is known to be exact when the input is a planar graph [17]. We note that this is similar to the LP formulation used by Gupta and Talwar [18] when specialized to the Min-Uncut problem, but their LP formulation is on the "label extended graph" that we will explain soon.



(a) The shortest path distance between any two pairs of vertices is 0. The bold edges correspond to an optimal integral solution to LP-MinUncut.

(b) After removing edges with weight at least 1/2 (the bold edges), all remaining subgraphs are of diameter at most 1/4 and they are bipartite. The dashed edges are the inter-cluster edges.

Figure 1 Applying low diameter graph decomposition in a feasible solution to LP-MinUncut.

$$\mathsf{LP}^\star = \min \sum_{e \in E} c_e x_e \qquad \text{(LP-MinUncut)}$$

subject to

$$\sum_{e \in C} x_e \geq 1 \quad C \in \mathcal{C}$$

$$x_e \geq 0 \quad e \in E$$

where $\mathcal{C}$ is the set of odd cycles of G.

This LP has exponentially many constraints. To solve it in polynomial time using the ellipsoid method [30], we require a polynomial time separation oracle to check whether a solution x is feasible or not, and if not provide a violating constraint. For this LP, it is well known that the separation oracle can be implemented in polynomial time using shortest path computations (e.g. see [18]). Since this will be relevant to our discussion, we describe the separation oracle in the following.

The idea is to construct the "label extended graph" $H = (V', E')$ of $G = (V, E)$ (to use the Unique Games terminology). For each vertex v in V, we create two vertices v^+ and v^- in V'. For each edge uv in E, we add two edges u^+v^- and u^-v^+ in E', and we set the weight of u^+v^- and u^-v^+ to be x_{uv}. By construction, there is an odd cycle in G containing v if and only if there is a path from v^+ to v^- in the label extended graph H. So, to check that x is feasible, we just need to check that the weight of the shortest path from v^+ to v^- is at least one for every v.

3.2 Proof Overview

One natural approach to do the rounding is to consider the label extended graph H of G. From the above discussion, destroying all the odd cycles in G is equivalent to destroying all the v^+-v^- paths in H for all v. Since x is feasible, we know that the shortest path distance between v^+ and v^- is at least 1 for every v. Therefore, we can apply the low diameter graph decomposition result in the label extended graph, by setting $\Delta < 1$ to ensure that all v^+ and v^- are disconnected, and hope to delete edges with weight at most $\sum_{e \in E} O(r/\Delta) \cdot c_e x_e = O(r) \cdot \mathsf{LP}^\star$ by Theorem 5. This is similar to the approach used in [35] to reduce Unique Games to Multicut. The problem of this approach is that the label extended graph H could have arbitrarily large clique minor, even though the original constraint graph G is K_r-minor free: Section 2 do not apply. even if the constraint graph G is grid-like and planar, the label-extended graph H can contain a $K_{\Omega(n)}$ minor, even when the alphabet size is just two. This means that applying the theorems in Section 2 blindly does not give better than a $\mathcal{O}(\log n)$-factor approximation.

This is often a technical issue in analyzing algorithms for Unique Games: It is most natural to work on the label extended graph but the label extended graph does not necessarily share the nice properties in the original graph [27]. It is not obvious how to apply low diameter graph decomposition directly in the original constraint graph to do the rounding. For example, in the graph shown in Figure 1a, x is an integral solution but the shortest path distance (using x_e as the edge weight of e) is 0 for all pairs of vertices, providing no useful information about which pairs of vertices we need to separate.

Our main observation is that the shortest path distances are not useful only when there are edges with large x_e. In Lemma 8, we prove that if $x_e < 1/2$ for every e, then every odd cycle contains a pair of vertices u, v with shortest path distance greater than $1/4$ (using x_e as the edge weight of e). Therefore, if we apply low diameter graph decomposition with $\Delta = 1/4$, then we can ensure that no odd cycle will remain in any cluster, and the above calculation shows that the total weight of the deleted edges is $\mathcal{O}(r) \cdot \mathsf{LP}^\star$. To reduce to the case where there are no edges with $x_e \geq 1/2$, we can simply delete all such edges as their total weight is at most $2\mathsf{LP}^\star$. This preprocessing step is remotely similar to some iterative rounding algorithms (see [28]). See Figure 1b for an illustration.

3.3 Rounding Algorithm

```
Algorithm 1 (Min-Uncut).
  Intput:  A feasible solution $x$ to LP-MinUncut with value $\mathsf{LP}^\star$ on a $K_r$-minor free graph.
  Output:  An integral solution to LP-MinUncut with total cost $O(r) \cdot \mathsf{LP}^\star$

1. Let $F_1$ be the subset of edges with $x_e \geq 1/2$. Delete all edges in $F_1$ from the graph.
2. Set the weight $w_e$ of each edge $e$ in the remaining graph to be $x_e$.
   Sample a weakly $(1/4)$-bounded $\mathcal{O}(r)$-separating partition $P$ guaranteed by Theorem 5
   in the remaining graph.
3. Let $F_2$ be the set of inter-cluster edges in $P$, i.e. edges $uv$ with $P(u) \neq P(v)$.
   Return $F_1 \cup F_2$ as the output.
```

3.4 Main Lemma

The following lemma allows us to apply low diameter graph decomposition in the original constraint graph. The proof uses the simple but crucial fact that if we "shortcut" an odd cycle, one of the two cycles created is an odd cycle.

▶ **Lemma 8.** *Let G' be a graph with edge weight x_e for each edge e. Suppose every odd cycle C has total weight at least 1, i.e. $\sum_{e \in C} x_e \geq 1$. If $0 \leq x_e < \delta \leq 1$ for every edge $e \in G'$, then every odd cycle C in G' contains a pair of vertices u, v satisfying $d_x(u, v) > (1 - \delta)/2$, where $d_x(u, v)$ denotes the shortest path distance from u to v induced by the edge weights x_e.*

Proof. Let C be an arbitrary odd cycle and let v_0 be an arbitrary vertex in C. We will prove the stronger statement that if $d_x(v_0, v) \leq (1 - \delta)/2$ for every $v \in C$, then there is an edge $e \in C$ with $x_e \geq \delta$. Note that the contrapositive of this stronger statement clearly implies the lemma.

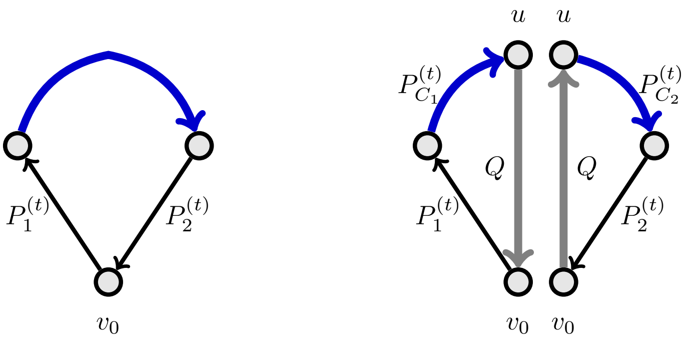


Figure 2 The paths involved in the proof of Lemma 8. The shortcut Q is highlighted gray, and the cycle segments $P_{C_j}^{(t)}$ are highlighted blue. Since the walk we maintain is odd, one of the two walks we consider in the induction step (right figure) should be odd.

Since all odd cycles have total weight at least 1, any nontrivial odd walk (may visit some vertices multiple times) from v_0 to v_0 has total weight at least 1. This is because any odd walk can be decomposed into edge-disjoint simple cycles, with at least one of which is odd.

We will prove the statement by an inductive argument. In a general inductive step $t \geq 0$, we maintain a walk $C^{(t)}$ from v_0 to v_0 satisfying the following properties (see Figure 2):

1. $C^{(t)}$ is a nontrivial odd walk from v_0 to v_0, consisting of three paths $P_1^{(t)}$-$P_C^{(t)}$-$P_2^{(t)}$.
2. $P_1^{(t)}$ and $P_2^{(t)}$ contain v_0, with v_0 being the first vertex of $P_1^{(t)}$ and v_0 being the last vertex of $P_2^{(t)}$.
3. Both $P_1^{(t)}$ and $P_2^{(t)}$ have total weight at most $(1-\delta)/2$.
4. $P_C^{(t)}$ is a continuous segment of C, i.e. if $C = (v_0, v_1, \dots, v_k = v_0)$, then $P_C^{(t)} = (v_i, \dots, v_j)$ for some $0 \leq i < j \leq k$. In particular, $P_C^{(t)} \neq \emptyset$.

Initially, $C^{(0)}$ is just the cycle C, with $P_1^{(0)} = P_2^{(0)} = \emptyset$ and $P_C^{(0)} = C$.

Let $w(P)$ denote the total weight of a path P, and let $|P|$ denote the number of edges in P. Since $w(P_1^{(t)}), w(P_2^{(t)}) \leq (1-\delta)/2$, we must have $w(P_C^{(t)}) \geq \delta$, as $C^{(t)}$ is a nontrivial odd walk and thus the total weight is at least one. The inductive step is to show that if $d_x(v_0, v) \leq (1-\delta)/2$ for all $v \in C$, then we can construct $C^{(t+1)}$ from $C^{(t)}$ so that $C^{(t+1)}$ still satisfies the properties but $|P_C^{(t+1)}| < |P_C^{(t)}|$. By applying this inductively, we will eventually construct a walk $C^{(T)}$ that satisfies the properties and $|P_C^{(T)}| = 1$, and so $P_C^{(T)}$ is an edge of C with weight $w(P_C^{(t)}) \geq \delta$, and this will complete the proof.

It remains to prove the inductive step (see Figure 2). Let $C^{(t)}$ be a walk that satisfies the properties but $|P_C^{(t)}| \geq 2$. Let u be an internal vertex of $P_C^{(t)}$, which splits $P_C^{(t)}$ into $P_{C_1}^{(t)}$ and $P_{C_2}^{(t)}$, so that the walk $C^{(t)}$ consists of $P_1^{(t)}$-$P_{C_1}^{(t)}$-$P_{C_2}^{(t)}$-$P_2^{(t)}$. Since $d_x(v_0, u) \leq (1-\delta)/2$, there is a path Q from v_0 to u with $w(Q) \leq (1-\delta)/2$. The path Q splits the walk $C^{(t)}$ into two walks, $P_1^{(t)}$-$P_{C_1}^{(t)}$-Q and Q-$P_{C_2}^{(t)}$-$P_2^{(t)}$ As $C^{(t)}$ is an odd walk, a simple parity argument implies that exactly one of these two walks must be odd, say $P_1^{(t)}$-$P_{C_1}^{(t)}$-Q (the other case is similar). Then we let $C^{(t+1)} := P_1^{(t)}$-$P_{C_1}^{(t)}$-Q, with $P_1^{(t+1)} := P_1^{(t)}$, $P_C^{(t+1)} := P_{C_1}^{(t)}$, and $P_1^{(t+1)} := Q$. It is straightforward to check that $C^{(t+1)}$ still satisfy all the properties and furthermore $|P_C^{(t+1)}| < |P_C^{(t)}|$, completing the proof of the induction step. ◀

3.5 Proof of Corollary 3

We are now ready to prove that the algorithm in Section 3.3 is an $O(r)$-approximation algorithm for Min-Uncut. In step 1, since each edge e in F_1 has $x_e \geq 1/2$, the total cost of edges in F_1 is

$$\sum_{e \in F_1} c_e \leq 2 \sum_{e \in F_1} c_e x_e \leq 2\mathsf{LP}^\star.$$

Let $G' := G - F_1$ be the remaining graph. By Lemma 8, every odd cycle of G' contains a pair of vertices u, v with shortest path distance greater than $1/4$. Let $\Delta = 1/4$. In a $(1/4)$-bounded partition P, no cluster can contain an odd cycle C as otherwise the pair of vertices $u, v \in C$ with $d_x(u, v) > 1/4$ guaranteed by Lemma 8 would contradict that the cluster has weak diameter at most $1/4$. So, each cluster induces a bipartite graph, and thus $G' - F_2$ is a bipartite graph where F_2 is the set of inter-cluster edges. Therefore, $F_1 \cup F_2$ is an integral solution to the Odd Cycle Transversal problem, and hence an integral solution to the Min-Uncut problem.

To complete the proof, it remains to bound the cost of the edges in F_2. We use Theorem 5 to sample from a distribution of partitions which is Δ-bounded and $\mathcal{O}(r)$-separating, and by definition (1) the probability of an edge e being an inter-cluster edge is at most $\mathcal{O}(r) \cdot x_e/\Delta = \mathcal{O}(r) \cdot x_e$. Therefore, the expected cost of F_2 is

$$\mathbb{E}\left[\sum_{e \in F_2} c_e\right] = \sum_{e=uv \in G'} c_e \cdot \mathbb{P}_{P \sim \mathcal{A}}[P(u) \neq P(v)] = \sum_{e \in G'} c_e \cdot \mathcal{O}(r) \cdot x_e = \mathcal{O}(r) \sum_{e \in G'} c_e x_e \leq \mathcal{O}(r) \cdot \mathsf{LP}^\star.$$

Hence, the expected total cost of edges in $F_1 \cup F_2$ is $\mathcal{O}(r) \cdot \mathsf{LP}^\star$, and this concludes the proof of Corollary 3 about K_r-minor free graphs. For bounded genus graphs, the proof is the same except that we use Theorem 7 which guarantees the partitioning scheme is $\mathcal{O}(\log g)$-separating.

4 Max-2Lin$_k$

In this section, we show that the Min-Uncut algorithm can be readily generalized to the Max-2Lin$_k$ problem. The proofs will be almost identical, so we just highlight the subtle differences.

One important feature of Theorem 2 is that the approximation ratio does not depend on the alphabet size. The reason is that the symmetry of the linear constraints allows us to define inconsistent cycles in the original constraint graph, which will play the same role as the odd cycles in the Min-Uncut problem. This allows us to reduce Max-2Lin$_k$ to the Inconsistent Cycle Transversal problem.

4.1 Problem Formulation

Consider the Max-2Lin$_k$ problem where each constraint is of the form $x_u - x_v = c_{uv} \pmod{k}$ where $c_{uv} \in \mathbb{Z}_k$. The symmetry property that we will exploit is that every permutation constraint π_{uv} satisfies: $\pi_{uv}(i+c) = \pi_{uv}(i)+c$ for all $i, c \in \mathbb{Z}_k$. Note that there are "directions" in the constraints, as $\pi_{uv} = (\pi_{vu})^{-1}$ and they are in general different. In the Max-Cut (or Min-Uncut) problems, we have $\pi_{uv} = \pi_{vu}$ as the alphabet set is of size two, and so the concept of direction was not discussed.

▶ **Definition 9** (Inconsistent cycles for Max-2Lin$_k$)**.** Let $\mathfrak{I} = (G, \Pi)$ be an instance of Max-2Lin$_k$. A cycle $(v_0, v_1, \ldots, v_l = v_0)$ of length l in G is called inconsistent if

$$\pi_{v_l v_{l-1}} \circ \pi_{v_{l-1} v_{l-2}} \circ \cdots \circ \pi_{v_1 v_0} \neq \mathrm{Id} \tag{2}$$

where Id is the identity permutation. By the aforementioned symmetry property of Max-2Lin$_k$, if the product π of permutation constraints along a cycle is not the identity permutation, then $\pi(i) \neq i$ for all $i \in \mathbb{Z}_k$. This is the crucial property that we will use.

The following lemma shows that Max-2Lin$_k$ is equivalent to the Inconsistent Cycle Transversal problem. The reason is that whether a cycle is satisfiable is independent of which label to assign to the starting vertex because of the symmetry property. Note that this does not hold for general Unique Games.

▶ **Lemma 10.** *A* Max-2Lin$_k$ *instance* $\mathfrak{I} = (G, \Pi)$ *is satisfiable if and only if* G *contains no inconsistent cycles.*

Proof. Suppose $\mathfrak{I}$ is satisfiable. Let x be a satisfying assignment. Consider an arbitrary cycle $C = (v_0, v_1, \ldots, v_l = v_0)$. The permutation constraints on C enforce that $\pi_{v_l v_{l-1}} \circ \pi_{v_{l-1} v_{l-2}} \circ \cdots \circ \pi_{v_1 v_0}(x(v_0)) = x(v_0)$ where $x(v_0)$ is the value of v_0 in the assignment x. By the symmetry property of the constraints, this implies that $\pi_{v_l v_{l-1}} \circ \pi_{v_{l-1} v_{l-2}} \circ \cdots \circ \pi_{v_1 v_0}$ is the identity permutation, and thus it is consistent.

Suppose G has no inconsistent cycles. Then we show that G is satisfiable by the following trivial algorithm. Pick an arbitrary vertex $v_0 \in G$, and set $x(v_0)$ an arbitrary value. Then we propagate this assignment to every other vertex v by using an arbitrary path $P = (v_0, v_1, \ldots, v_l = v)$ from v_0 to v and set $x(v) = \pi_{v_l v_{l-1}} \circ \pi_{v_{l-1} v_{l-2}} \circ \cdots \circ \pi_{v_1 v_0}(v_0)$. In particular, we can use a breadth first search tree to propagate the assignment. Since G has no inconsistent cycles, any two paths P_1, P_2 from v_0 to v will define the same value $x(v)$, as otherwise following P_1 from v_0 to v and following P_2 from v to v_0 will give us an inconsistent cycle. This implies that any non-tree constraint uv is also satisfied by the assignment, as otherwise it means that there are two paths from v_0 to v defining different values from $x(v)$, one path being the tree path from v_0 to u plus the edge uv, and the other path being the tree path from v_0 to v. ◀

4.2 Linear Programming Relaxation

Given Lemma 10, we can formulate the minimization version of the Max-2Lin$_k$ problem, the Min-2Lin$_k$ problem, as the Inconsistent Cycle Transversal problem, where the objective is to find a subset of edges of minimum cost that intersects all the inconsistent cycles. We can then use the same linear programming relaxation for the Min-Uncut problem, with $\mathcal{C}$ being the set of inconsistent cycles in the constraint graph. Again, we can design a polynomial time separation oracle to check whether a solution x is feasible, by constructing the label extended graph and using shortest path computations as in Section 3.1 (see [18]).

4.3 Rounding Algorithm and Analysis

The rounding algorithm is exactly the same as in Section 3.3, and so we do not repeat it here. The analysis is also the same, which relies on a generalization of Lemma 8.

▶ **Lemma 11.** *Let* G' *be a graph with edge weight* x_e *for each edge* e*. Suppose every inconsistent cycle* C *has total weight at least* 1*, i.e.* $\sum_{e \in C} x_e \geq 1$*. If* $0 \leq x_e < \delta \leq 1$ *for every edge* $e \in G'$*, then every inconsistent cycle* C *in* G' *contains a pair of vertices* u, v *satisfying* $d_x(u, v) > (1 - \delta)/2$*, where* $d_x(u, v)$ *denotes the shortest path distance from* u *to* v *induced by the edge weights* x_e*.*

Proof. The proof is essentially identical, by replacing every occurrence of "odd" by "inconsistent". The only place that needs explanation is in the last paragraph of Lemma 8, when we split an inconsistent walk using a path Q from v_0 to u into two walks $P_1^{(t)}$-$P_{C_1}^{(t)}$-Q and Q-$P_{C_2}^{(t)}$-$P_2^{(t)}$, and we need to argue that at least one of these two walks is inconsistent. Suppose both walks are consistent. Let π_{P_1} be the composition of the permutation constraints from v_0 to u following the path $P_1^{(t)}$-$P_{C_1}^{(t)}$, π_Q be the composition of the permutation constraints from u to v_0 following the path Q, and π_{P_2} be the composition of the permutation constraints from u to v_0 following the path $P_{C_2}^{(t)}$-$P_2^{(t)}$. The first walk is consistent means that $\pi_Q \circ \pi_{P_1} = \mathrm{Id}$, and the second walk is consistent means that $\pi_{P_2} \circ (\pi_Q)^{-1} = \mathrm{Id}$. But this implies that following the first walk and then the second walk is consistent, and thus the original walk is also consistent as $\mathrm{Id} = (\pi_{P_2} \circ (\pi_Q)^{-1}) \circ (\pi_Q \circ \pi_{P_1}) = \pi_{P_2} \circ \pi_{P_1}$, contradicting that the original walk is inconsistent. The rest of the proof is identical. ◀

With Lemma 11, using exactly the same argument as in Section 3.5 gives us the proof of Theorem 2.

5 General Unique Games

For general Unique Games, we could not reduce the problem to some cycle cutting problem in the original constraint graph. Instead, we modify the LP-based algorithm of Gupta and Talwar [18] to prove Theorem 4.

5.1 Linear Programming Relaxation

Gupta and Talwar [18] use the following linear programming relaxation for the Unique Games problem.

$$\min \mathsf{LP}^\star = \sum_{uv \in E} \frac{c_{uv}}{2} \sum_{l=1}^{k} d(u,v,l) \qquad \text{(LP-UG)}$$

subject to

$$\sum_{l=1}^{k} x(u,l) = 1 \qquad \forall u \in V$$

$$d(u,v,l) \geq |x(u,l) - x(v,\pi_{uv}(l))| \qquad \forall uv \in E,\ l \in [k]$$

$$\sum_{i=1}^{t} d(v_{i-1}, v_i, l_{i-1}) \geq x(u, l_0) \qquad \forall C,\ \forall u \in C,\ \forall l_0 \in B_{u,C}$$

$$1 \geq x(u,l) \geq 0 \qquad \forall u \in V,\ \forall l \in [k]$$

The intended value of $x(u,l)$ is 1 if we assign the label l to vertex u and 0 otherwise, and so the first constraint enforces that we assign exactly one label to each vertex. The intended value of $d(u,v,l)$ is 1 if we assign u to l but not assign v to $\pi_{uv}(l)$ or vice versa and is 0 otherwise. So $\sum_{l=1}^{k} d(u,v,l)$ is two if the constraint π_{uv} is not satisfied and is 0 if the constraint is satisfied, and therefore the objective function is to minimize the total cost of the violated constraints. The third constraint is the inconsistent cycle constraint in the label extended graph: $B_{u,C}$ is defined as the set of "bad" labels at u, so that if u is assigned some label in $B_{u,C}$, then propagating this label along the cycle must violate some permutation constraint in C. So, the intention of the third constraint is that if we assign some label in $B_{u,C}$ to vertex u, then the number of violated constraint along the cycle C must be at least 1. This is similar to our inconsistent cycle constraint, but defined on the label extended graph.

5.2 Proof Overview

Gupta and Talwar [18] gave a polynomial time randomized algorithm to return an integral solution of cost $\mathcal{O}(\log n) \cdot \mathsf{LP}^\star$ from a feasible solution to the LP with objective value $\mathsf{LP}^\star$.

The main technique in their rounding algorithm is the use of a low average distortion tree to propagate an assignment from a vertex. Their propagation rounding algorithm picks an arbitrary vertex $u \in V$ and assigns it a random label l_u according to the probability distribution defined by $x(u,l)$. Then they design a correlated sampling scheme to sample a label l_v for a neighbor v of u satisfying the properties that $\mathbb{P}[l_v = l] = x(v,l)$ and $\mathbb{P}[l_v \neq \pi_{uv}(l_u)] \leq \sum_{l=1}^{k} d(u,v,l)$. They use this correlated sampling to propagate the assignment from the starting vertex to every vertex in the graph using the low average distortion tree. Their approximation ratio comes from the average distortion $\mathcal{O}(\log n)$ of the tree given by the FRT embedding [15], which can not be improved even for planar graphs.

We will still use the propagation rounding method of Gupta and Talwar, but we apply it to different trees. In [18], the tree T needs not be a spanning tree in the constraint graph (i.e. some edges in the tree may not exist in the graph), and this adds some complication to the analysis. In our application, all tree edges will be graph edges and we can use a simpler lemma in their proof. For an edge $uv \in E$, we let $d_G(u,v) := \sum_{l=1}^{k} d(u,v,l)$, and let $d_T(u,v) := \sum_{xy \in P} d_G(x,y)$ where P is the unique path from u to v in the tree T.

▶ **Lemma 12** (Lemma 3.1 in [18]). *Let x be the assignment produced by the propagation rounding algorithm using correlated sampling along a tree T. For every edge $uv \in G$, we have*

$$\mathbb{P}[x(v) \neq \pi_{uv}(x(u))] \leq d_G(u,v) + 2d_T(u,v).$$

The idea of our algorithm is very simple. We use the strongly Δ-bounded $\mathcal{O}(r^2)$-separating partitioning scheme to decompose the graph, using $d_G(u,v)$ as the weight of edge $uv \in E(G)$. As each cluster is of strong diameter Δ, we simply use a shortest path tree in each cluster to do the propagation rounding and apply Lemma 12 to prove Theorem 4. We will choose Δ to balance the losses in the two steps.

5.3 Rounding Algorithm

```
Algorithm 2 (UG_k).
  Intput:  A feasible solution $x, d$ to LP-UG with value $\mathsf{LP}^\star$ on a $K_r$-minor free graph.
  Output:  An integral solution to LP-UG with total cost $O(r) \cdot \sqrt{\mathsf{LP}^\star}$.

1. Set the weight $w_{uv}$ of each edge $uv$ to be $d_G(u,v)$.
   Sample a strongly $\Delta$-bounded $\mathcal{O}(r^2)$-separating partition $P$ guaranteed by Theorem 6.
2. Let $F$ be the set of inter-cluster edges in $P$, i.e. edges $uv$ with $P(u) \neq P(v)$.
   Delete $F$ from $G$.
3. In each cluster $C_j$ in the remaining graph, compute a shortest path tree $T_j$.
4. Run Gupta-Talwar propagation rounding on each cluster $C_j$ using tree $T_j$.
5. Return the solution $x, d$ as the union of the solution in each cluster.
```

5.4 Proof of Theorem 4

Since the partitioning scheme is $\mathcal{O}(r^2)$-separating, by definition (1), each edge e is deleted with probability

$$\mathbb{P}[\text{edge } uv \text{ is deleted}] = \mathcal{O}(r^2) \cdot \frac{d_G(u,v)}{\Delta}.$$

Hence, the expected total cost of the deleted edges in Step 2 is

$$\sum_{uv\in E} c_{uv} \cdot \mathbb{P}[\text{edge } uv \text{ is deleted}] = \mathcal{O}(r^2/\Delta) \sum_{uv\in E} c_{uv} \cdot d_G(u,v) = \mathcal{O}(r^2/\Delta) \cdot \mathsf{LP}^\star.$$

We just assume that all of these edges will be violated by the assignment we produce at the end. Since each cluster C_j has strong diameter Δ, the shortest path tree T_j satisfies

$$d_{T_j}(u,v) \le \Delta \quad \forall u,v \in C_j.$$

Using the Gupta-Talwar propagation rounding, by Lemma 12, each edge in cluster C_j is violated with probability $O(\Delta)$, and therefore the total cost of the violating constraints in the Step 4 is at most $O(\Delta)\sum_{e\in E} c_e$. By choosing $\Delta = r \cdot \sqrt{\mathsf{LP}^\star / \sum_{e\in E} c_e}$, the total cost of the violating constraints is at most $r \cdot \sqrt{\mathsf{LP}^\star \cdot \sum_{e\in E} c_e}$. When $\mathsf{LP}^\star = \varepsilon \cdot \sum_{e\in E} c_e$, the total cost of the violating constraint is at most $r\sqrt{\varepsilon}\sum_{e\in E} c_e$, proving Theorem 4 for K_r-minor free graphs. For bounded genus graphs, we just use the bound in Theorem 7 to replace r^2 by $\log g$, and the same proof gives Theorem 4 for bounded genus graphs.

6 Discussions and Open Problems

The algorithm for general Unique Games has a similar structure to the subexponential time algorithm [5]. Both algorithms first deletes a small fraction of edges so that each remaining component has some nice properties, and then solve the problem in each component using a propagation rounding method. The nice property in [5] is that each component has few small eigenvalues (which qualitatively means that the components have good expansion property), and the decomposition result is based on random walks. The nice property in this paper is that each component has small diameter, and the decomposition result is based on some combinatorial methods. The key to these algorithms is some graph decomposition result. Is there some property that captures both good expansion and small diameter so that graph decomposition is still possible? Is there some property that captures both good expansion and small diameter so that propagation rounding still works?

Another open question is whether the ideas in this paper can be generalized to handle graphs with many small eigenvalues.

Acknowledgements. We thank Tsz Chiu Kwok, Akshay Ramachandran and Hong Zhou for useful discussions.

References

1 Ittai Abraham, Cyril Gavoille, Anupam Gupta, Ofer Neiman, and Kunal Talwar. Cops, robbers, and threatening skeletons: Padded decomposition for minor-free graphs. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*. ACM, 2014.

2 Amit Agarwal, Moses Charikar, Konstantin Makarychev, and Yury Makarychev. O(sqrt(log n)) approximation algorithms for min uncut. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, 2005.

3 Naman Agarwal. Unique games conjecture: The boolean hypercube and connections to graph lifts. Master's thesis, University of Illinois at Urbana-Champaign, 2014.

4 Naman Agarwal, Guy Kindler, Alexandra Kolla, and Luca Trevisan. Unique games on the hypercube. *Chicago Journal of Theoretical Computer Science*, 2015, 2015.

5 Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2010.

6 Sanjeev Arora, Subhash A. Khot, Alexandra Kolla, David Steurer, Madhur Tulsiani, and Nisheeth K. Vishnoi. Unique games on expanding constraint graphs are easy. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*. ACM, 2008.

7 Brenda S. Baker. Approximation algorithms for np-complete problems on planar graphs. *Journal of the ACM*, 41, 1994.

8 Nikhil Bansal, Uriel Feige, Robert Krauthgamer, Konstantin Makarychev, Viswanath Nagarajan, Joseph Naor, and Roy Schwartz. Min-max graph partitioning and small set expansion. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2011.

9 Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2011.

10 Punyashloka Biswal, James R. Lee, and Satish Rao. Eigenvalue bounds, spectral partitioning, and metrical deformations via flows. *Journal of the ACM*, 57, 2010.

11 Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Near-optimal algorithms for unique games. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*. ACM, 2006.

12 Eden Chlamtac, Konstantin Makarychev, and Yury Makarychev. How to play unique games using embeddings. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2006.

13 Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Ken ichi Kawarabayashi. Algorithmic graph minor theory: Decomposition, approximation, and coloring. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, 2005.

14 Erik D. Demaine and MohammadTaghi Hajiaghayi. The bidimensionality theory and its algorithmic applications. *The Computer Journal*, 51, 2008.

15 Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*. ACM, 2003.

16 Jittat Fakcharoenphol and Kunal Talwar. *An improved decomposition theorem for graphs excluding a fixed minor*. Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques. Springer Berlin Heidelberg, 2003.

17 Jean Fonlupt, Ali Ridha Mahjoub, and J. P. Uhry. Compositions in the bipartite subgraph polytope. *Discrete mathematics*, 105, 1992.

18 Anupam Gupta and Kunal Talwar. Approximating unique games. In *Proceedings of the 70th Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2006.

19 Venkatesan Guruswami and Ali Kemal Sinop. Lasserre hierarchy and approximation schemes for graph partitioning and quadratic integer programming with psd objectives. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2011.

20 Frank Hadlock. Finding a maximum cut of a planar graph in polynomial time. *SIAM Journal on Computing*, 4, 1975.

21 Jonathan A. Kelner. Spectral partitioning, eigenvalue bounds, and circle packings for graphs of bounded genus. *SIAM Journal on Computing*, 35, 2006.

22 Jonathan A. Kelner, James R. Lee, Gregory N. Price, and Shang-Hua Teng. Metric uniformization and spectral bounds for graphs. *Geometric and Functional Analysis*, 21, 2011.

23 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 34th Annual ACM Symposium on Theory of computing*. ACM, 2002.

24 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, 2007.

25 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-ε. *Journal of Computer and System Sciences*, 74, 2008.

26 Philip Klein, Serge A. Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. In *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, 1993.

27 Alexandra Kolla. Spectral algorithms for unique games. *Computational Complexity*, 20, 2011.

28 Lap Chi Lau, Ramamoorthi Ravi, and Mohit Singh. *Iterative methods in combinatorial optimization*, volume 46. Cambridge University Press, 2011.

29 James R. Lee and Anastasios Sidiropoulos. Genus and the geometry of the cut graph. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2010.

30 Lászlo Lovász, Martin Grötschel, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*. Springer-Verlag, Berlin, 1988.

31 Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*. ACM, 2008.

32 Prasad Raghavendra and David Steurer. Graph expansion and the unique games conjecture. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*. ACM, 2010.

33 Prasad Raghavendra, David Steurer, and Madhur Tulsiani. Reductions between expansion problems. In *Proceedings of the 27th IEEE Annual Conference on Computational Complexity*. IEEE, 2012.

34 Daniel A. Spielman and Shang-Hua Teng. Spectral partitioning works: Planar graphs and finite element meshes. *Linear Algebra and its Applications*, 421, 2007.

35 David Steurer and Nisheeth K. Vishnoi. Connections between unique games and multicut. *Electronic Colloquium on Computational Complexity*, 16, 2009.

36 Luca Trevisan. Approximation algorithms for unique games. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2005.

Greedy Minimization of Weakly Supermodular Set Functions

Edo Liberty[1] **and Maxim Sviridenko**[2]

1 **Amazon, Inc., New York, NY, USA**
libertye@amazon.com

2 **Yahoo! Research, New York, NY, USA**
sviri@yahoo-inc.com

Abstract

This paper defines *weak-α-supermodularity* for set functions. It shows that minimizing such functions under cardinality constrains is a common task in machine learning and data mining. Moreover, any problem whose objective function exhibits this property benefits from a greedy extension phase. Explicitly, let S^* be the optimal set of cardinality k that minimizes f and let S_0 be an initial solution such that $f(S_0) \leq \rho f(S^*)$. Then, a greedy extension $S \supset S_0$ of size $|S| \leq |S_0| + \lceil \alpha k \ln(\rho/\varepsilon) \rceil$ yields $f(S) \leq (1+\varepsilon) f(S^*)$.

Example usages of this framework give streamlined proofs and new bi-criteria results for k-means, sparse regression, column subset selection, and sparse convex function minimization. Sparse regression and column subset selection are special cases of a new, more general, sparse multiple linear regression problem that is of independent interest. This paper also corrects a brittleness of the proof of Natarajan for the properties of the greedy algorithm for sparse regression.

1998 ACM Subject Classification G.1.3 Numerical Linear Algebra, G.1.6 Optimization, G.4 Mathematical Software

Keywords and phrases Weak Supermodularity, Greedy Algorithms, Machine Learning, Data Mining

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.19

1 Introduction

Many problems in data mining and unsupervised machine learning take the form of minimizing a set function with cardinality constraints. More explicitly, denote by $[n]$ the set $\{1, \ldots, n\}$ and $f(S) : 2^{[n]} \to \mathbb{R}_+$. Our goal is to minimize $f(S)$ subject to $|S| \leq k$. These problems include clustering and covering problems as well as sparse regression, matrix approximation problems and many others. These combinatorial problems are hard to minimize in general. Finding good (e.g. constant factor) approximate solutions for them requires significant sophistication and highly specialized algorithms.

In this paper we analyze the behavior of the greedy algorithm to all of these problems. We start by claiming that the functions above are special. A trivial observation is that they are non-negative and non-increasing, that is, $f(S) \geq f(S \cup T) \geq 0$ for any $S, T \subseteq [n]$. This immediately shows that expanding solution sets is (at least potentially) beneficial in terms of reducing the function value. But, monotonicity is not sufficient to ensure that any number of greedy extensions of a given solution would significantly reduce the objective function.

To this end we need to somehow quantify the gain of adding a single element (greedily) to a solution set. Let $f(S) - f(S \cup T)$ be the reduction in f one gains by adding a set

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 19; pp. 19:1–19:11
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

of elements T to the current solution S. Then, the average gain of adding elements from T *sequentially* is $[f(S) - f(S \cup T)]/|T \setminus S|$. One would hope that there exists an element in $i \in T \setminus S$ such $f(S) - f(S \cup \{i\}) \geq [f(S) - f(S \cup T)]/|T \setminus S|$. However, that would be false in general because different element contributions are not independent of each other. Nevertheless, it is true for supermodular functions (see Fact 3).

Combining this fact with the idea that T could be any set, including the optimal solution S^*, already gives some useful results for minimizing supermodular set functions. Specifically those for which $f(S^*)$ is bounded away from zero. Notice that k-means clustering (defined below) is exactly this kind of problem. Section 4 gives some new bicriteria results obtainable for k-means via the greedy extension algorithm of Section 3.

Alas, most problems of interest, such as regression, column subset selection, and feature selection are not supermodular. In Section 2 we define the notion of weak-α-supermodularity. Intuitively, weak-α-supermodular functions are those conducive to greedy type algorithms. The property requirers that there exists an element $i \in T \setminus S$ such that adding i *first* gains at least $[f(S) - f(S \cup T)]/\alpha|T \setminus S|$ for some $\alpha \geq 1$.

An analogous relaxation of the submodular property for set functions was considered in [5] (see definition 2.3). They define a *submodularity-ratio* for set functions which are not submodular. They show that if the *submodularity-ratio* is bonded, a greedy algorithm can be used to obtain bi-crateria results for the maximization problem. The work of [5] can be viewed as a direct extension of well know fact. Namely that the greedy algorithm provides a $(1 - 1/e)$-factor approximation for maximizing set functions $g(S)$ subject to $|S| \leq k$ if g for positive, monotone non-decreasing and submodular set functions [15].

This paper complements both [15] and [5] for the intuitively related process of greedily minimizing supermodular functions. While our setting is not significantly more complex it is quite different. In contrast to maximizing submodular functions, minimizing supermodular functions is, in general, hard [10]. The difficulty arrises from the fact that a value of zero of the objective function could force any constant factor approximation algorithm to find an optimal solution. Our work cannot overcome this fundamental (and unresolvable) difficulty.

We consider the case where either the objective function is bounded away from zero or one could obtain an approximate initial solution. In that case, supermodularity (or weak-α-supermodularity) is shown to be sufficient for obtaining good bi-crateria results using the greedy algorithm. Section 2 includes notations and concepts that will be used throughout the paper. In section 3 we present two generic greedy algorithms and analyze their guaranties for weak-α-supermodular functions.

Many important problems in data mining and machine learning fall into this regime. As a warm-up, in Section 4 we obtain new bi-creteria results for k-means clustering, the objective function of which is supermodular. Section 5 presents the sparse multiple linear regression (SMLR) and shows that it is weakly-α-supermodular. We then streamline and slightly improve the result of [14] for sparse regression, also known as feature selection. Column Subset Selection (CSS) for matrix approximation is an instance of SMLR. Section 7 gives new bi-creteria results for CSS with little additional effort. Finally, we recreate the result of [16] for minimizing smooth and strongly convex functions with sparse solutions. The result is equivalent but the proof is simpler and shorter.

2 Preliminaries and definitions

Throughout the manuscript we denote by $[n]$ the set $\{1, \ldots, n\}$. We concern ourselves with non-negative set function $f(S) : 2^{[n]} \to \mathbb{R}_+$. More specifically monotone non-increasing set function such that $f(S) \geq f(S \cup T)$ for any two sets $S \subseteq [n]$ and $T \subseteq [n]$.

Algorithm 1 Greedy Extension Algorithm

```
Input: Weakly-$\alpha$-supermodular function $f(S)$, initial set $S_0$, parameters $k \in \mathbb{Z}_+$ and the sequence $\Lambda_1, \Lambda_2, \ldots$
while $t \leq \lceil \alpha k \ln \Lambda_t \rceil$ do
    $S_t \leftarrow S_{t-1} \cup \arg\min_{i \in [n]} f(S_{t-1} \cup \{i\})$
Output: $S_t$
```

▶ **Definition 1.** A set function $f(S) : 2^{[n]} \to \mathbb{R}_+$ is said to be *supermodular* if for any two sets $S, T \subseteq [n]$

$$f(S \cap T) + f(S \cup T) \geq f(S) + f(T). \tag{1}$$

▶ **Definition 2.** A non-negative non-increasing set function $f(S) : 2^{[n]} \to \mathbb{R}_+$ is said to be *weakly-α-supermodular* if there exists $\alpha \geq 1$ such that for any two sets $S, T \subseteq [n]$

$$f(S) - f(S \cup T) \leq \alpha \sum_{i \in T \setminus S} (f(S) - f(S \cup \{i\})) . \tag{2}$$

This property is useful because we will later try to minimize f. It asserts that if adding $T \setminus S$ is beneficial then there is an element $i \in T \setminus S$ that contributes at least a fraction of that. The reason for the name of this property might also be explained by the following definition and lemma.

▶ **Fact 3.** *A non-increasing non-negative supermodular function f is weakly-α-supermodular with parameter $\alpha = 1$.*

Proof. For $S, T \subseteq [n]$ order the set $T \setminus S$ in an arbitrary order, i.e. $T \setminus S = \{i_1, \ldots, i_{|T \setminus S|}\}$. Define $R_0 = \emptyset$ and $R_t = \{i_1, \ldots i_t\}$ for $t > 0$. By supermodularity we have for any t

$$f(S) - f(S \cup \{i_t\}) \geq f(S \cup R_{t-1}) - f(S \cup R_{t-1} \cup \{i_t\}) \tag{3}$$

We note that $R_{t-1} \cup \{i_t\} = R_t$ and sum up Equation (3).

$$\begin{aligned} \sum_{t=1}^{|T \setminus S|} [f(S) - f(S \cup \{i_t\})] &\geq \sum_{t=1}^{|T \setminus S|} f(S \cup R_{t-1}) - f(S \cup R_{t-1} \cup \{i_t\}) \\ &= f(S) - f(S \cup T) \ . \end{aligned}$$

Since $|T \setminus S| \cdot \max_{i \in T \setminus S}[f(S) - f(S \cup \{i\})] \geq \sum_{t=1}^{|T \setminus S|}[f(S) - f(S \cup \{i_t\})]$ this implies weak-1-supermodularily. ◀

3 General Greedy Extension Algorithms

We are given a weakly-α-supermodular set function $f(S)$ and would like to solve the following optimization problem

$$\min\{f(S) : |S| \leq k\}. \tag{4}$$

Let $0 < \Lambda_1 \leq \Lambda_2 \leq \ldots$ be a non-decreasing bounded sequence of reals, i.e. $\max_t \Lambda_t < +\infty$. Our algorithm works in phases and we may assume that Λ_t is computed on step t of the algorithm. Consider a simple greedy algorithm that starts with some initial solution S_0 of value $f(S_0)$ (maybe $S_0 = \emptyset$) and sequentially and greedily adds elements to it to minimize f.

Note that since the sequence Λ_t is bounded the algorithm terminates after at most $\lceil \alpha k \ln (\max_t \Lambda_t) \rceil$ iterations.

Algorithm 2 Greedy Extension Algorithm

```
**Input:** Weakly-$\alpha$-supermodular function $f(S)$, initial set $S_0$, $k \in \mathbb{Z}_+$
**while** $t \leq \lceil \alpha k \ln \left( f(S_0)/\varepsilon f(S_{t-1}) \right) \rceil$ **do**
    $S_t \leftarrow S_{t-1} \cup \arg\min_{i \in [n]} f(S_{t-1} \cup \{i\})$
**Output:** $S_t$
```

▶ **Lemma 4.** *Let S_τ be the output of Algorithm 1. Then $|S_\tau| \leq |S_0| + \lceil \alpha k \ln \Lambda_\tau \rceil$ and $f(S_\tau) \leq f(S^*) + \frac{f(S_0) - f(S^*)}{\Lambda_{\tau+1}}$ where S^* is an optimal solution of the optimization problem* (4)*.*

Proof. The fact that $|S_\tau| \leq |S_0| + \lceil \alpha k \ln \Lambda_\tau \rceil$ is a trivial observation. For the second claim consider an arbitrary iteration $t \in [\tau]$ and consider the set $S^* \setminus S_{t-1}$. By monotonicity and weak α-supermodularity

$$\begin{aligned} f(S_{t-1}) - f(S^*) &\leq f(S_{t-1}) - f(S_{t-1} \cup S^*) \\ &\leq \alpha \cdot \sum_{i \in S^* \setminus S_{t-1}} f(S_{t-1}) - f(S_{t-1} \cup \{i\}) \\ &\leq \alpha k \cdot \max_{i \in [n]} f(S_{t-1}) - f(S_{t-1} \cup \{i\}) \\ &= \alpha k \cdot \left(f(S_{t-1}) - f(S_t) \right) . \end{aligned}$$

By rearranging the above equation and recursing over t we get

$$f(S_t) - f(S^*) \leq \left(f(S_{t-1}) - f(S^*) \right) (1 - 1/\alpha k) \leq \left(f(S_0) - f(S^*) \right) (1 - 1/\alpha k)^t$$

Substituting $\tau + 1 > \lceil \alpha k \ln \Lambda_{\tau+1} \rceil$ for the last step of the algorithm completes the proof.

$$\begin{aligned} f(S_\tau) - f(S^*) &\leq \left(f(S_0) - f(S^*) \right) (1 - 1/\alpha k)^{\alpha k \ln \Lambda_{\tau+1}} \\ &\leq \left(f(S_0) - f(S^*) \right) e^{-\ln \Lambda_{\tau+1}} \leq \frac{f(S_0) - f(S^*)}{\Lambda_{\tau+1}} . \end{aligned}$$ ◀

▶ **Theorem 5.** *Let S_τ be the output of Algorithm 2 which is an instantiation of Algorithm 1 with parameters $\Lambda_t = f(S_0)/\varepsilon f(S_{t-1})$ for some error $\varepsilon \geq 0$. Then $|S_\tau| \leq |S_0| + \lceil \alpha k \ln(f(S_0)/\varepsilon f(S^*)) \rceil$ and $f(S_\tau) \leq f(S^*)/(1 - \varepsilon)$ where S^* is an optimal solution of the optimization problem* (4)*.*

Proof. By Lemma 4 we have

$$|S_\tau| \leq |S_0| + \lceil \alpha k \ln \Lambda_\tau \rceil \leq |S_0| + \lceil \alpha k \ln(f(S_0)/\varepsilon f(S^*)) \rceil$$

and

$$\begin{aligned} f(S_\tau) &\leq f(S^*) + \frac{f(S_0) - f(S^*)}{\Lambda_{\tau+1}} \\ &= f(S^*) + \frac{f(S_0) - f(S^*)}{f(S_0)} \varepsilon f(S_\tau) \leq f(S^*) + \varepsilon f(S_\tau). \end{aligned}$$ ◀

▶ **Theorem 6.** *Assume there exist a ρ-approximation algorithm creating S_0 such that $f(S_0) \leq \rho f(S^*)$. There exists an algorithm for generating S such that $|S| \leq |S_0| + \lceil \alpha k \left(\ln \frac{\rho}{\varepsilon} \right) \rceil$ and $f(S) \leq f(S^*)/(1 - \varepsilon)$.*

Proof. Use the ρ-approximation algorithm to create S_0 for Algorithm 1 and apply Theorem 5. ◀

Algorithm 3 Greedy Extension Algorithm; an alternative stopping criterion

```
Input: Weakly-$\alpha$-supermodular function $f$, $S_0$, $f_{\text{stop}}$
repeat
    $S_t \leftarrow S_{t-1} \cup \arg\min_i f(S_{t-1} \cup \{i\})$
until $f(S_t) \le f_{\text{stop}}$
Output: $S = S_t$
```

▶ **Theorem 7.** *Let k' be the minimal cardinality of a set S' such that $f(S') \le f'$. For any f_{stop} and an initial set S_0 such that $f' < f_{\text{stop}} < f(S_0)$ Algorithm 3 outputs S such that*

$$|S| \le |S_0| + \left\lceil \alpha k' \left(\ln \frac{f(S_0) - f'}{f_{\text{stop}} - f'} \right) \right\rceil$$

Proof. Let $f' = f(S')$. The proof follows from Lemma 4 by setting $k = k_f$, $\Lambda_t = \frac{f(S_0)-f'}{f_{\text{stop}}-f'}$ and noticing that $\frac{f(S_0)-f'}{f_{\text{stop}}-f'} \le \frac{f(S_0)-f}{f_{\text{stop}}-f}$. ◀

This alternative algorithm will be used in Section 6

4 k-means Clustering

As a gentle introduction we begin with deriving new bi-cretiria results for the k-means clustering problem. We begin by defining the constrained k means problem.

▶ **Definition 8** (Constrained k-means). Given a set of n points $X \subset \mathbb{R}^d$, find a set $S \subset X$ minimizing $f(S) = \sum_{x \in X} \min_{x' \in S} \|x - x'\|^2$ subject to $|S| \le k$.

▶ **Lemma 9.** *For the constrained k-means problem, one can find in $O(n^2 dk \log(1/\varepsilon))$ time a set S of size $|S| = O(k) + k \log(1/\varepsilon)$ such that $f(S) \le (1+\varepsilon) f(S^*)$ where $f(S^*)$ is the optimal solution.*

Proof. The constrained k-means objective function f is weakly-1-supermodular because it is supermodular (Fact 3). This is both well known and not hard to reverify. Using the algorithm of [1] one obtains a set S_0 of size $|S_0| = O(k)$ points from X for which $f(S_0) = O(f(S^*))$. Their technique improves on the analysis of well known k-means++ adaptive sampling scheme of [2]. Greedily extending S_0 and applying the analysis of Theorem 5 completes the proof. The quadratic dependency of the running time on the number of data points can be alleviated using the corset construction of [8, 9] ◀

▶ **Definition 10** (Unconstrained k-means). Given a set of n points $X \subset \mathbb{R}^d$, find a set $S \subset \mathbb{R}^d$ minimizing $f(S) = \sum_{x \in X} \min_{c \in S} \|x - c\|^2$ subject to $|S| \le k$.

▶ **Lemma 11.** *Let $f(S^*)$ be the optimal solution to the unconstrained k-means problem. One can find in time $O(n^2 dk \log(1/\varepsilon))$ a set $S \in \mathbb{R}^d$ of size $|S| = O(k) + k \log(1/\varepsilon)$ such that $f(S) \le (2+\varepsilon) f(S^*)$.*

Proof. The proof and the algorithm are identical to the above. The only point to note is that a $1 + \varepsilon/2$ approximation to the constrained problem is at most a $2 + \varepsilon$ approximation to the unconstrained one. See [2], for example, for the argument that the minimum of the constrained objective is at most twice that of the unconstrained one. ◀

Alternatively, we can utilize a more computationally expensive approach which goes through a reduction to the k-median problem.

▶ **Definition 12** (k-Median)**.** We are given a set X of data points, the set $\mathcal{C}$ of potential cluster center locations and the nonnegative costs $w_{ij} \geq 0$ for all $i, j \in X \times \mathcal{C}$. Find a set $S \subset \mathcal{C}$ minimizing $f(S) = \sum_{i \in X} \min_{j \in \mathcal{C}} w_{ij}$ subject to $|S| \leq k$.

It is known that given an instance (X, k) of the Unconstrained k-means problem one can construct in polynomial time an instance of the k-Median problem $(X, \mathcal{C}, w, k)$ where $\mathcal{C} \subseteq \mathbb{R}^d$ such that for any solution of value Φ for the Unconstrained k-means problem there exists a solution of value $(1+\varepsilon)\Phi$ for the corresponding instance of the k-Median problem (see Theorem 7 [13]). Moreover, $|\mathcal{C}| = n^{O(\log(1/\varepsilon)/\varepsilon^2)}$. Therefore, after applying this transformation on our instance of the Unconstrained k-means and using the same initial solution S_0 as in Lemma 11 we derive.

▶ **Lemma 13.** *Let $f(S^*)$ be the optimal solution to the unconstrained k-means problem. One can find in time $O(n^{O(\log(1/\varepsilon)/\varepsilon^2)}dk)$ a set $S \in \mathbb{R}^d$ of size $|S| = O(k) + k\log(1/\varepsilon)$ such that $f(S) \leq (1+\varepsilon)f(S^*)$.*

5 Sparse Multiple Linear Regression

We begin by defining the Sparse Multiple Linear Regression (SMLR) problem. Given two matrices $X \in \mathbb{R}^{m \times n}$ and $Y \in \mathbb{R}^{m \times \ell}$, and an integer k find a matrix $W \in \mathbb{R}^{n \times \ell}$ that minimizes $\|XW - Y\|_F^2$ subject to W having at most k non zero rows. We assume for notational brevity (and w.l.o.g.) that the columns of X have unit norm. An alternative and equivalent formulation of SMLR is as follows. Let X_S be a submatrix of the matrix X defined by the columns of X indexed by the set $S \subseteq [n]$. Let X_S^+ be the Moore-Penrose pseudo-inverse of X_S. It is well-known that the minimizer of $\|XW - Y\|_F^2$ subject to W whose non zero rows are indexed by S is equal to $\|Y - X_S X_S^+ Y\|_F^2$. SMLR can therefore be reformulated as

$$\min_{S \subseteq [n]} \{f(S) = \|Y - X_S X_S^+ Y\|_F^2 : |S| \leq k\} \ .$$

We can consequently apply our methodology from Section 3 to SMLR if we show that $f(S)$ is α-weakly-supermodular.

▶ **Lemma 14.** *For $X \in \mathbb{R}^{m \times n}$ and $Y \in \mathbb{R}^{m \times \ell}$ the SMLR minimization function $f(S) = \|Y - X_S X_S^+ Y\|_F^2$ is α-weakly-supermodular with $\alpha = \max_{S'} \|X_{S'}^+\|_2^2$.*

Proof. We first estimate $f(S) - f(S \cup T)$. Denote by $Z_{T \setminus S}$ the matrix whose columns are those of $X_{T \setminus S}$ projected away from the span of X_S and normalized. More formally, $\zeta_i = \|(I - X_S X_S^+)x_i\|$ and $z_i = (I - X_S X_S^+)x_i/\zeta_i$ for all $i \in T \setminus S$. Note that the column span of $Z_{T \setminus S}$ is orthogonal to that of X_S and that together they are equal to the column span of $X_{T \cup S}$. Using the Pythagorean theorem and the fact that $X_S X_S^+$ is a projection we obtain $f(S) = \|Y\|_F^2 - \|X_S X_S^+ Y\|_F^2$ and $f(S \cup T) = \|Y\|_F^2 - \|X_S X_S^+ Y\|_F^2 - \|Z_{S \setminus T} Z_{S \setminus T}^+ Y\|_F^2$. Substituting $T = \{i\}$ also gives $f(S) - f(S \cup \{i\}) = \|z_i z_i^T Y\|_F^2 = \|z_i^T Y\|_2^2$.

$$\begin{aligned} f(S) - f(S \cup T) &= \|Z_{T \setminus S} Z_{T \setminus S}^+ Y\|_F^2 && (5) \\ &= \|(Z_{T \setminus S}^T)^+ \cdot Z_{T \setminus S}^T Y\|_F^2 \quad \text{By SVD} && (6) \\ &\leq \|(Z_{T \setminus S}^T)^+\|_2^2 \cdot \|Z_{T \setminus S}^T Y\|_F^2 && (7) \\ &= \|Z_{T \setminus S}^+\|_2^2 \cdot \sum_{i \in T \setminus S} \|z_i^T Y\|_2^2 && (8) \end{aligned}$$

$$\begin{aligned} &\le \|X^+_{T\cup S}\|_2^2 \cdot \sum_{i\in T\setminus S} \|z_i^T Y\|_2^2 \quad \text{See below} && (9)\\ &= \alpha \cdot \sum_{i\in T\setminus S} (f(S) - f(S\cup\{i\})) && (10)\end{aligned}$$

For Equation (9) we use a non trivial transition, $\|Z^+_{T\setminus S}\|_2 \le \|X^+_{T\cup S}\|_2$. By the definition of $Z_{T\setminus S}$ we can write for $i \in T\setminus S$ that $z_i = (x_i - \sum_{j\in S}\alpha_{ij}x_j)/\zeta_i$ and $\zeta_i = \|(I - X_S X_S^+)x_i\|$. For any vector w

$$Z_{T\setminus S} w = \sum_{i\in T\setminus S} x_i w_i/\zeta_i + \sum_{j\in S} x_j \sum_{i\in T\setminus S} w_i\alpha_{ij}/\zeta_i = X_{T\cup S}w'$$

where $w'_i = w_i/\zeta_i$ for $i \in T\setminus S$ and $w'_j = \sum_{i\in T\setminus S} w_i\alpha_{ij}/\zeta_i$ for $j \in S$. Since, $\zeta_i = \|(I - X_S X_S^+)x_i\| \le \|x_i\| = 1$ we have $\|w'\| \ge \|w\|$. Finally, consider w such that $\|w\| = 1$ and $\|Z_{T\setminus S}w\| = \|Z^+_{T\setminus S}\|^{-1}$. This is the right singular vector corresponding to the smallest singular value of $Z_{T\setminus S}$. We obtain

$$\|Z^+_{T\setminus S}\|^{-1} = \|Z_{T\setminus S}w\| = \|X_{T\cup S}w'\| \ge \|X^+_{T\cup S}\|^{-1}\|w'\| \ge \|X^+_{T\cup S}\|^{-1} \ .$$

This completes the proof. ◀

▶ **Lemma 15.** *Let $f(S^*)$ be the optimal solution to the Sparse Multiple Linear Regression problem. One can find in time $O(\alpha k \log(\|Y\|_F^2/\varepsilon)\cdot nT_f)$ a set $S \subseteq [n]$ of size $|S| = \lceil \alpha k \log(\|Y\|_F^2/\varepsilon)\rceil$ such that $f(S) \le f(S^*)/(1-\varepsilon)$ where T_f is the time needed to compute $f(S)$ once.*

6 Sparse Regression

The problem of Sparse Regression defined in [14] is an instance of SMLR where the number of columns in W and Y is $\ell = 1$. Since both W and Y are vectors we reduce to the more familiar form of this problem; minimize $\|Xw - y\|_2^2$ subject to $\|w\|_0 \le k$.

Natarajan [14] analyzes the greedy algorithm for the sparse regression problem. He sets a desired threshold error E and defines k to be the minimum cardinality of a solution S^* that achieves $f(S^*) \le E' = E/4$. He shows that for $\alpha = \max_{S'} \|X^+_{S'}\|^2$ the greedy algorithm finds a solution S such that $f(S) \le E$ such that

$$|S| \le \left\lceil 9k\alpha \ln \frac{\|y\|_2^2}{E}\right\rceil \ .$$

Natarajan's implicit assumption

In [14] Natarajan uses $\alpha = \|X^+\|^2$ instead of $\alpha = \max_{S'}\|X^+_{S'}\|^2$. This is only correct if the columns of X are linearly independent which seems to be an implicit assumption. In this setting $\alpha = \max_{S'}\|X^+_{S'}\|^2 = \|X^+\|^2$ by Cauchy's interlacing theorem. Note that $\max_{S'}\|X^+_{S'}\| \ge \|X^+\|$ if the columns of X are linearly dependent. This is the setting in the hardness result of [10] and is inevitable in the under constrained case where the number of columns is larger than their dimension.

Here, we apply Theorem 7 with initial solution $S_0 = \emptyset$ (which gives $f(S_0) = \|y\|_2^2$) and $E' = E/4$. It immediately yields that the greedy algorithm finds a solution of value $f(S) \le E$ and

$$|S| \le \left\lceil k\alpha \ln \frac{\|y\|_2^2 - E/4}{E - E/4}\right\rceil \le \left\lceil \frac{4}{3}k\alpha \ln \frac{\|y\|_2^2}{E}\right\rceil$$

using the inequality $\ln(\frac{4}{3}x - \frac{1}{3}) \leq \frac{4}{3}\ln x$ for $x \geq 1$. This improves the result of [14] in three ways

1. the approximation constant is smaller
2. its proof is more streamlined and
3. it extends to viability of the greedy algorithm to the under constrained case where the result of [14] does not hold.

7 Column Subset Selection Problem

Given a matrix X, Column Subset Selection (CSS) is concerned with finding a small set of columns whose span captures as much of the Frobenius norm of X. It was throughly investigated in the context of numerical linear algebra [11, 12, 4]. CSS can be formulated as follows, find a subset $S \in [n]$, $|S| \leq k$ of matrix columns that minimize $f(S) = \|X - X_S X_S^+ X\|_F^2$. This formulation makes it clear that this is a special case of SMLR where $Y = X$.

The work of [17] investigates the notion of a curvature $c \in [0, 1]$ for a nonincreasing set functions. They define it as follows:

$$c = 1 - \min_{j\in[n]} \min_{S,T\subseteq[n]\setminus\{j\}} \frac{f(S) - f(S \cup \{j\})}{f(T) - f(T \cup \{j\})}. \tag{11}$$

They show that there exists a greedy type algorithm that finds a solution of value at most $1/(1 - c)$ times the optimal value of the minimization problem for any objective set function with curvature c (Corollary 8.5 in [17]).

▶ **Lemma 16** (Lemma 9.1 from [17])**.** *Let $f(S)$ be the objective function for the Column Subset Selection Problem corresponding to the matrix X. The curvature c of $f(S)$ is such that $\frac{1}{1-c} \leq \kappa^2(X)$ where $\kappa(X)$ is the condition number of X.*

Note that for any matrix $X$with full column rank if $\tilde{X}$ is the matrix with normalized columns then $\|\tilde{X}^+\| \leq \kappa(X)$. We can find our initial solution S_0 by one of the three known methods:

1. an approximation algorithm from [17] finds a solution S_0 such that $|S_0| = k$ and performance guarantee $\rho = \kappa^2(X)$;
2. an approximation algorithm from [7, 6] with $|S_0| = k$ and $\rho = k + 1$;
3. an approximation algorithm from [3] with $|S_0| = 2k$ and $\rho = 2$;

▶ **Lemma 17.** *For the column subset selection problem for a column normalized matrix X and $\alpha = \max_{S'} \|X_{S'}^+\|_2^2$ one can find a set S such that*

$$f(S) \leq (1 + \varepsilon)f(S^*) \qquad \textit{and} \qquad |S| = O(k) + \alpha k \left(\ln \frac{\rho}{\varepsilon}\right).$$

Proof. Combining one of the above results with the algorithm from Section 3 completes the proof. ◀

8 Sparse Convex Function Minimization

One popular extension of the regression problem is to consider

$$f(S) = \min\{R(y) : \operatorname{supp}(y) \subseteq S\} \tag{12}$$

where $R(y)$ is a convex function and $\operatorname{supp}(y) = \{i \mid y_i \neq 0\}$. Following Shalev-Shwartz et al. [16], we consider a special case when the convex function $R(y)$ satisfies two additional conditions.

▶ **Definition 18.** A function $R(w)$ is said to be λ-strongly convex for $\lambda \geq 0$ if for each $w, u \in \mathbb{R}^d$ we have

$$R(w) \geq R(u) + \langle \nabla R(u), w - u \rangle + \frac{\lambda}{2}||w - u||_2^2.$$

▶ **Definition 19.** A function $R(w)$ is said to be β-smooth if for each $w, u \in \mathbb{R}^d$ we have

$$R(w) \leq R(u) + \langle \nabla R(u), w - u \rangle + \frac{\beta}{2}||w - u||_1^2.$$

Shalev-Shwartz et al. [16] gave many examples of such convex functions. In particular, they relate our Definition 19 to a class of functions arising in Machine Learning with β-smooth loss functions (see Lemma B1 and Section 3 in [16]).

▶ **Theorem 20.** *Given the set function $f(S)$ defined in (12) corresponding to β-smooth λ-strongly convex function $R(w)$. The set function $f(S)$ is α-weakly-supermodular with $\alpha = \frac{\beta}{\lambda}$.*

Proof. Let $y_S \in \mathbb{R}^d$ be a vector minimizing the function $R(y)$ among vectors with support S and $y_{S\cup T} \in \mathbb{R}^d$ be a vector minimizing function $R(y)$ among vectors with support $S \cup T$. For any vector $x \in \mathbb{R}^d$, let $x(j) \in \mathbb{R}$ be its j-th coordinate.

For each $j \in T \setminus S$, we define vector $\tilde{y}^j \in \mathbb{R}^d$ such that $\tilde{y}^j(j) = y_{S\cup T}(j)$ and $\tilde{y}^j(i) = 0$ for all $i \neq j$. It is enough to prove the inequality

$$R(y_S) - R(y_{S\cup T}) \leq \frac{\beta}{\lambda} \sum_{j\in T\setminus S} R(y_S) - R\left(y_S + \frac{\lambda}{\beta}\tilde{y}^j\right) \tag{13}$$

to prove the statement of the theorem. By applying Definitions 18 and 19 we derive

$$\begin{aligned}
\sum_{j\in T\setminus S} R(y_S) - R\left(y_S + \frac{\lambda}{\beta}\tilde{y}^j\right) &\geq \sum_{j\in T\setminus S} \left(-\left\langle \nabla R(y_S), \frac{\lambda}{\beta}\tilde{y}^j\right\rangle - \frac{\beta}{2}||\frac{\lambda}{\beta}\tilde{y}^j||_1^2\right) \\
&\geq -\frac{\lambda}{\beta}\left(\sum_{j\in T\setminus S} \langle \nabla R(y_S), \tilde{y}^j \rangle\right) - \frac{\lambda^2}{2\beta}||y_{S\cup T} - y_S||_2^2 \\
&= -\frac{\lambda}{\beta}\langle \nabla R(y_S), y_{S\cup T} - y_S\rangle - \frac{\lambda^2}{2\beta}||y_{S\cup T} - y_S||_2^2 \\
&\geq \frac{\lambda}{\beta}\left(R(y_S) - R(y_{S\cup T}) + \frac{\lambda}{2}||y_{S\cup T} - y_S||_2^2\right) - \frac{\lambda^2}{2\beta}||y_{S\cup T} - y_S||_2^2 \\
&= \frac{\lambda}{\beta}\left(R(y_S) - R(y_{S\cup T})\right)
\end{aligned}$$

where the first equality follows from the fact that $\nabla R(y_S)(j) = 0$ for all $j \in S$. ◀

Let R^* be the target value for our convex function $R(y)$ and k_f be the minimal cardinality of a set S' such that $f(S') \leq R^*$ where $f(S)$ is defined by (12). Combining Theorem 7 and Theorem 20 we derive

▶ **Theorem 21.** *For any $\varepsilon > 0$, let $f_{\text{stop}} = R^* + \varepsilon$ then the Algorithm 3 outputs S such that*

$$|S| \leq \left\lceil \frac{\beta}{\lambda} k_f \left(\ln \frac{R(\emptyset) - R^*}{\varepsilon}\right)\right\rceil.$$

The above theorem is analogous to Theorem 2.8 in [16].

Acknowledgments. We would like to thanks Christos Boutsidis for his contributions to early drafts of this paper, Sergei Vassilvitskii and Dan Feldman for their guidance and Petros Drineas for pointing out that Natarajan's proof would potentially not carry through for column subset selection.

References

1 Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for k-means clustering. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 12th International Workshop, APPROX 2009, and 13th International Workshop, RANDOM 2009, Berkeley, CA, USA, August 21-23, 2009. Proceedings*, pages 15–28, 2009. doi:10.1007/978-3-642-03685-9_2.

2 David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *SODA*, pages 1027–1035, 2007.

3 C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014.

4 T. F. Chan and P. C. Hansen. Some applications of the rank revealing qr factorization. *SIAM Journal on Scientific and Statistical Computing*, 13:727, 1992.

5 A. Das and D. Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In *In Proceedings of ICML*, pages 1057–1064, 2011.

6 A. Deshpande and L. Rademacher. Efficient volume sampling for row/column subset selection. In *Proceedings of the 42th Annual ACM Symposium on Theory of Computing (STOC)*, 2010.

7 Amit Deshpande, Luis Rademacher, Santosh Vempala, and Grant Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2:225–247, 2006.

8 Dan Feldman, Amos Fiat, Micha Sharir, and Danny Segev. Bi-criteria linear-time approximations for generalized k-mean/median/center. In *Proceedings of the Twenty-third Annual Symposium on Computational Geometry*, SCG'07, pages 19–26, New York, NY, USA, 2007. ACM. doi:10.1145/1247069.1247073.

9 Dan Feldman and Michael Langberg. A unified framework for approximating and clustering data. In *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing*, STOC'11, pages 569–578, New York, NY, USA, 2011. ACM. doi:10.1145/1993636.1993712.

10 Dean P. Foster, Howard J. Karloff, and Justin Thaler. Variable selection is hard. In *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*, pages 696–709, 2015.

11 G. H. Golub. Numerical methods for solving linear least squares problems. *Numer. Math.*, 7:206–216, 1965.

12 M. Gu and S. C. Eisenstat. Efficient algorithms for computing a strong efficient algorithms for computing a strong rank-revealing qr-factorization. *SIAM Journal on Scientific Computing*, 17(848–869), 1996.

13 K. Makarychev, Y. Makarychev, M. Sviridenko, and J. Ward. A bi-criteria approximation algorithm for k means. In *submission*, 2015.

14 B.K. Natarajan. Sparse approximate solutions to linear systems. *SIAM J. Comput.*, 24(2):227–234, April 1995. doi:10.1137/S0097539792240406.

15 G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions – i. *Mathematical Programming*, 14(1):265–294, 1978. doi:10.1007/BF01588971.

16 S. Shalev-Shwartz, N. Srebro, and T. Zhang. Trading accuracy for sparsity in optimization problems with sparsity constraints. *SIAM Journal on Optimization*, 20(6):2807–2832, 2010.

17 Maxim Sviridenko, Jan Vondrak, and Justin Ward. Optimal approximation for submodular and supermodular optimization with bounded curvature. *In Proceedings of SODA 2015*, pages 1134–1148, 2014.

Renyi Entropy Estimation Revisited*

Maciej Obremski[1] and Maciej Skorski[2]

1 Aarhus University, Aarhus, Denmark†
obremski@cs.au.dk

2 IST Austria, Klosterneuburg, Austria‡
maciej.skorski@gmail.com

Abstract

We revisit the problem of estimating entropy of discrete distributions from independent samples, studied recently by Acharya, Orlitsky, Suresh and Tyagi (SODA 2015), improving their upper and lower bounds on the necessary sample size n. For estimating Renyi entropy of order α, up to constant accuracy and error probability, we show the following

- Upper bounds $n = O(1) \cdot 2^{\left(1-\frac{1}{\alpha}\right)H_\alpha}$ for integer $\alpha > 1$, as the worst case over distributions with Renyi entropy equal to H_α.
- Lower bounds $n = \Omega(1) \cdot K^{1-\frac{1}{\alpha}}$ for any real $\alpha > 1$, with the constant being an inverse polynomial of the accuracy, as the worst case over all distributions on K elements.

Our upper bounds essentially replace the alphabet size by a factor exponential in the entropy, which offers improvements especially in low or medium entropy regimes (interesting for example in anomaly detection). As for the lower bounds, our proof explicitly shows how the complexity depends on both alphabet and accuracy, partially solving the open problem posted in previous works.

The argument for upper bounds derives a clean identity for the variance of falling-power sum of a multinomial distribution. Our approach for lower bounds utilizes convex optimization to find a distribution with possibly worse estimation performance, and may be of independent interest as a tool to work with Le Cam's two point method.

1998 ACM Subject Classification G.1.2 Approximation, G.3 Statistical Computing

Keywords and phrases Renyi entropy, entropy estimation, sample complexity, convex optimization

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.20

1 Introduction

1.1 Renyi Entropy

Renyi entropy [25] arises in many applications as a generalization of Shannon Entropy [27]. It is also of interests on its right, with a number of applications including unsupervised learning (like clustering) [30, 12], multiple source adaptation [17], image processing [16, 20, 26], password guessability [3, 24, 10], network anomaly detection [15], quantifying neural activity [22] or to analyze information flows in financial data [13].

In particular Renyi entropy of order 2, known also as collision entropy, is used in quality tests for random number generators [14, 29], to estimate the number of random bits

[*] Available at http://eprint.iacr.org/2017/588.pdf

[†] This project has received funding from the European research Council (ERC) under the European Unions's Horizon 2020 research and innovation programme (grant agreement No 669255).

[‡] Supported by the European Research Council consolidator grant (682815-TOCNeT).

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 20; pp. 20:1–20:15
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Algorithm 1: Estimation of Renyi Entropy

```
Input: entropy parameter $\alpha > 1$ (integer),
       alphabet $\mathcal{A} = \{a_1, \ldots, a_K\}$,
       samples $x_1, \ldots, x_n$ from an unknown distribution $p$ on $\mathcal{A}$
Output: number $H$ approximating the $\alpha$-entropy of $p$
$I \leftarrow \{i : \exists j \quad a_i = x_j\}$          /* compute the list of occurring symbols[1] */
for $i \in I$ do
    $n_i \leftarrow \#\{j : x_j = a_i\}$          /* compute empirical frequencies */
end
$M \leftarrow \sum_i \frac{n_i^{\underline{\alpha}}}{n^{\underline{\alpha}}}$          /* bias-corrected power sum estimation by falling powers[2] */
$H \leftarrow \frac{1}{1-\alpha} \log M$          /* entropy from power sums */
return $H$
```

that can be extracted from a physical source [11, 7], characterizes security of certain key derivation functions [4, 8], helps testing graph expansion [9] and closeness of distributions to uniformity [6, 23] and bounds the number of reads needed to reconstruct a DNA sequence [19].

1.2 Estimation and Sample Complexity

Motivated by the discussed applications, algorithms that estimate Renyi entropy of an unknown distribution from samples were proposed for discrete [31] and also for continuous distributions [21]. For Shannon entropy, estimators with multiplicative errors were studied in [5] and follow-up works; the existence of sublinear (in terms of the alphabet size) additive estimators was proved in [22], and the optimal additive estimator was given in [28]. For the general case of Renyi entropy, the state of the art was established in [1], with upper and lower bounds on the sample complexity.

Interestingly, the estimation of Renyi entropy of integer orders $\alpha \geqslant 1$ is *sublinear* in the alphabet size. More precisely, to estimate the entropy of an integer order $\alpha > 1$ of a distribution over an alphabet of size K, with a *constant accuracy and constant error probability*, one needs

$$n = \Theta(K^{1-\frac{1}{\alpha}})$$

samples. On the other hand, the necessary sample size for non-integer $\alpha > 1$ is

$$n = \Omega(K^{1-o(1)}),$$

with the upper bound $O(K/\log K)$, for large K and the accuracy sufficiently small [1, 2].

The estimator itself is a biased-reduced adaptation of the naive "plug-in" estimator. Note that computing empirical frequencies as estimates to true probabilities and putting them straight into the entropy formula (which we refer to as naive estimation) would yield a biased estimator. To obtain better convergence properties, one needs to add some *corrections* to the formula. In the case of Renyi entropy, one replaces powers of empirical frequencies in the entropy formula by *falling powers*, obtaining better estimator with the complexity bounds discussed above [1]. See Algorithm 1 for the pseudocode.

[2] Storing and updating empirical frequencies can be implemented with different data structures, we don't discuss the optimal solution as our primary interest is in the sample complexity.

[2] Here $z^{\underline{\alpha}}$ stands for the falling α-power of the number z.

1.3 Our contribution

1.3.1 Results

We revisit the analysis of the minimal number of samples n (sample complexity) needed to estimate Renyi entropy up to certain additive accuracy, obtaining improvements upon the result in [1]. In the presentation below we consider the estimation up to constant error probability, unless stated otherwise.

(a) Better upper bounds for the sample complexity, with a simplified analysis:

$$n = O\left(2^{\left(1-\frac{1}{\alpha}\right)H_\alpha}\delta^{-2}\right), \quad \text{for integer } \alpha > 1$$

valid for Algorithm 1, any accuracy $\delta > 0$, and all distributions with Renyi entropy of order α equal to H_α

(b) Lower bounds for non-integer $\alpha > 1$, explicit w.r.t. both alphabet and accuracy:

$$n = \Omega(1) \cdot \max\left(\delta^{-\frac{1}{\alpha}}K^{1-\frac{1}{\alpha}}, \delta^{-\frac{1}{2}}K^{\frac{1}{2}}\right), \qquad \text{for any non-integer } \alpha > 1$$

valid for any estimator, any accuracy $\delta \leqslant 1$ and some distribution over K elements.

(c) Refining the technique for proving lower bounds; we explain how to obtain optimal bounds for the ideas used in [1]; our construction for lower bounds is also simpler.

The first improvement essentially parameterizes the previous bound by the entropy amount, and is of interest in *medium/low entropy* regimes. Note that when the entropy is at most a half of the maximal amount ($H_\alpha \leqslant \frac{1}{2}\log K$) then the complexity drops to $n = O(K^{\frac{1}{2}})$ even for most demanding min-entropy ($\alpha = \infty$). The improvements may be relevant for anomaly detection algorithms based on evaluating entropy of data streams [15]. The precise statement, which addresses arbitrary accuracy and error probability, appears in Corollary 7.

The lower bounds given in [1] and improved in the journal version [2] depend only on the alphabet, and are valid for large K and sufficiently small δ. As opposed to that, our lower bounds apply to all regimes of K and δ and explicitly show that *large alphabets and small accuracy both contribute to the complexity*. Thus we make a progress[3] towards understanding how the sample complexity depends on δ and K, which is an open problem except for integer α [2]. In particular, our results show that the sample complexity may be much bigger than $\Omega\left(K^{1-o(1)}\right)$ for δ being small depending on K, which is not guaranteed by the previous results (e.g. Table 1 in [2]).

The technique for lower bound in [1] essentially boils down to the construction of two statistically close distributions that differ in entropy (the technique known as *Le Cam's two-point method*). The authors obtained implicitly a suboptimal pair with this property. We instead construct explicitly a simpler pair with much better properties.

1.3.2 Techniques

The original proof of the upper bounds proceeds by estimating the variance of the falling-power sum in Line 5 in Algorithm 1. This analysis is somewhat difficult because the empirical frequencies n_i in Line 3 are not independent. A workaround proposed in [1] uses *Poisson sampling* to randomize the number n in a convenient way (which doesn't hurt the convergence

[3] Our result is worse in the dependency on K, but the added value is the dependency on δ.

Table 1 Our lower bounds for estimation of Renyi entropy of order α. By K we denote the alphabet size, δ is the additive error of estimation, $\Omega(1)$ is an absolute constant.

Entropy	Accuracy	Sample Complexity
$1 < \alpha < 2$	$\delta \leqslant 1$	$\Omega(1) \cdot \min\left(\delta^{-\frac{1}{2}} K^{\frac{1}{2}}, \delta^{-\alpha} K^{1-\frac{1}{\alpha}}\right)$
	$\delta > 1$	$\Omega(1) \cdot \min\left(\left(2^{-\delta} K\right)^{\frac{1}{2}}, 2^{-\left(1-\frac{1}{\alpha}\right)\delta} K^{1-\frac{1}{\alpha}}\right)$
$2 \leqslant \alpha$	$\delta \leqslant 1$	$\Omega(1) \cdot \delta^{-\frac{1}{\alpha}} K^{1-\frac{1}{\alpha}}$
	$\delta > 1$	$\Omega(1) \cdot \left(2^{-\left(1-\frac{1}{\alpha}\right)\delta} K\right)^{1-\frac{1}{\alpha}}$

much), so that the frequencies are independent and the variance of power sums can directly computed.

We get rid of the Poisson sampling, by showing that the falling-power sum obeys a nice and clean algebraic identity, that can be further used to compute the variance (see Lemma 4). We believe that our technique may be of benefit to related problems, e.g. when estimating moments for streaming algorithms.

The argument for lower bounds in [1] starts by modifying the estimator so that it is a function of empirical frequencies (called *profiles* in [1]). Then, by certain facts on zeros of polynomials and exponential sums, one exhibits two probability distributions with certain relations between power sums. As a conclusion, again under Poisson sampling, one obtains two distributions such that their profiles differ much in entropy, yet are close in total variation. This yields a contradiction unless n is big enough.

Our approach deviates from these techniques. We share the same core idea, that estimation should be continuous in total variation, yet use it to conclude a clear bound without referring to profiles: if distributions are γ-close and the entropy differs by δ, the number n must satisfy $n = \Omega(\gamma^{-1})$ (see Corollary 9). It remains to construct two such distributions with possibly small γ and possibly big δ. By solving the related optimization task (which we do by an elegant application of *majorization theory*), we conclude that a simpler and better choice is one distribution being flat, and other being a combination of a flat distribution with a unit mass (see the proof of Lemma 11). We remark that our optimization approach not only gives better lower bounds for Renyi entropy, but may be also applied to similar estimation problems, e.g. lower bounds on the complexity for estimating functionals of a discrete distribution. The lower bounds are summarized in Table 1.

2 Preliminaries

For any natural α and real number x, by $x^{\underline{\alpha}} \overset{def}{=} \prod_{i=0}^{\alpha-1}(x-i)$ we denote the α-th falling power of x, with the convention $x^{\underline{0}} = 1$. If a discrete random variable X has a probability distribution p, we denote $p(x) = \Pr[X = x]$. For any distribution X by X^n we denote the n-fold product of independent copies of X. The moment of a distribution p of order α equals $p_\alpha = \sum_x p(x)^\alpha$. Through the paper, we use logarithms at base 2.

▶ **Definition 1** (Total variation (statistical closeness)). For two distributions p, q over the same finite alphabet the total variation equals $d_{TV} = \frac{1}{2}\sum_x |p(x) - q(x)|$. If $d_{TV}(p, q) \leqslant \epsilon$ we also say that p and q are ϵ-close.

▶ **Definition 2** (Renyi Entropy). The Renyi entropy of order α for $\alpha > 1$ equals

$$H_\alpha(p) \overset{def}{=} -\frac{1}{\alpha-1}\log\left(\sum_x p(x)^\alpha\right) = -\frac{1}{\alpha-1}\log p_\alpha.$$

Sometimes for shortness we simply say "α-entropy", referring to Renyi entropy of order α.

▶ **Definition 3** (Entropy Estimators). Given an alphabet $\mathcal{X}$ and a fixed number n we say that an algorithm $\hat{f}$ provides a (δ,ϵ)-approximation to α-entropy if for any distribution p over $\mathcal{X}$

$$|\hat{f}(x_1,\ldots,x_n) - H_\alpha(p)| > \delta$$

holds with probability at most ϵ over samples $x_1,\ldots,x_n$ drawn independently from p.

3 Auxiliary Facts

Define $\xi_i(x) = [X_i = x]$ and the *empirical frequency* of the symbol x by

$$n(x) = \sum_{i=1}^{n} \xi_i(x). \tag{1}$$

Note that the vector $(n(x))_{x\in\mathcal{X}}$ follows a multinomial distribution with sum n and probabilities $(p(x))_{x\in\mathcal{X}}$. The lemma below states that we have very simple expressions for the falling powers of $n(x)$.

▶ **Lemma 4** (Falling powers of empirical frequencies). *For every x we have*

$$n(x)^{\underline{\alpha}} = \sum_{i_1\neq i_2\neq\ldots\neq i_\alpha} \xi_{i_1}(x)\xi_{i_2}(x)\cdot\ldots\cdot\xi_{i_\alpha}(x). \tag{2}$$

In particular, we have

$$\mathbb{E}\left[\sum_x n(x)^{\underline{\alpha}}\right] = n^{\underline{\alpha}} p_\alpha. \tag{3}$$

The proof appears in Appendix A. We also obtain the following closed-form expressions for the variance of the sum of falling powers.

▶ **Lemma 5** (Variance of frequency falling powers sums). *We have*

$$\operatorname{Var}\left[\sum_x n(x)^{\underline{\alpha}}\right] = n^{\underline{\alpha}}((n-\alpha)^{\underline{\alpha}} - n^{\underline{\alpha}})p_\alpha^2 + \sum_{\ell=1}^{\alpha} n^{\underline{\alpha}}(n-\alpha)^{\underline{\alpha-\ell}}\binom{\alpha}{\ell}^2 l!\; p_{2\alpha-\ell}. \tag{4}$$

The proof appears in Appendix B.

4 Upper Bounds

Similarly as in [1], we observe that to estimate Renyi entropy with additive accuracy $O(\delta)$, it suffices to estimate power sums with multiplicative accuracy $O(\delta)$.

▶ **Theorem 6** (Estimator Performance). *The number of samples needed to estimate p_α up to a multiplicative error δ and error probability ϵ equals $n = O_\alpha\left(2^{\frac{\alpha-1}{\alpha}\cdot H_\alpha(p)}\delta^{-2}\log(1/\epsilon)\right)$.*

From this result one immediately obtains

▶ **Corollary 7.** *The number of samples needed to estimate $H_\alpha(p)$, up to an additive error δ and error probability ϵ, equals $n = O_\alpha\left(2^{\frac{\alpha-1}{\alpha}\cdot H_\alpha(p)}\delta^{-2}\log(1/\epsilon)\right)$. The matching estimator is Algorithm 1.*

Proof of Theorem 6. It suffices to construct an estimator with error probability $\frac{1}{3}$. We can amplify this probability to ϵ with a loss of a factor of $O(\log(1/\epsilon))$ in the sample size, by a standard argument: running the estimator in parallel on fresh samples and taking the median (as in [1]).

From Lemma 5 we conclude that the variance of the estimator equals

$$\mathrm{Var}[\mathsf{Est}] = -\Theta_\alpha(1)\cdot n^{-1}(p_\alpha)^2 + \sum_{\ell=1}^{\alpha}\Theta_\alpha(1)\cdot n^{-\ell}p_{2\alpha-\ell},$$

where $\Theta_\alpha(1)$ are constants dependent on α. Note that we have

$$p_{2\alpha-\ell} \leqslant (p_\alpha)^{\frac{2\alpha-\ell}{\alpha}}$$

by elementary inequalities[4], and therefore

$$\mathrm{Var}[\mathsf{Est}] = O_\alpha(1)\cdot p_\alpha^2\sum_{\ell=1}^{\alpha}\left(np_\alpha^{\frac{1}{\alpha}}\right)^{-\ell} = O_\alpha(1)\cdot n^{-1}p_\alpha^{2-\frac{1}{\alpha}}\sum_{\ell=0}^{\alpha-1}\left(np_\alpha^{\frac{1}{\alpha}}\right)^{-\ell}.$$

Note that the negative term $-\Theta_\alpha(1)n^{-1}(p_\alpha)^2$ we skipped is of smaller order than the term $\ell = 1$ of the sum on the right hand side, so it doesn't help to improve the bounds. For $n > 2p_\alpha^{\frac{1}{\alpha}}$ the right hand side equals $O_\alpha(1)\cdot n^{-1}p_\alpha^{1-\frac{1}{\alpha}}$. By the Chebyszev Inequality

$$\Pr_{X^n\sim p}\left[|\mathsf{Est}(X^n)) - p_\alpha| > \delta p_\alpha\right] < \frac{\mathrm{Var}[\mathsf{Est}]}{\delta^2 p_\alpha^2} = O_\alpha(1)\cdot n^{-1}p_\alpha^{-\frac{1}{\alpha}}\delta^{-2},$$

which is smaller than $\frac{1}{3}$ for some $n = O_\alpha(1)\cdot p_\alpha^{-\frac{1}{\alpha}}\delta^{-2}$. ◀

5 Lower Bounds

We will need the following lemma, stated in a slightly different way in [1]. It captures the intuition that if two distributions differ much in entropy, then they must be far away in total variation (otherwise the estimator, presumably working well, would distinguish them).

▶ **Lemma 8** (Estimation is continuous in total variation). *Suppose that $\hat{f}$ is a (δ,ϵ)-estimator for H_α. Then the following is true:*

$$\forall X, Y \quad |H_\alpha(X) - H_\alpha(Y)| \geqslant 2\delta \Rightarrow d_{TV}(X^n; Y^n) \geqslant 1 - 2\epsilon. \tag{5}$$

The proof is illustrated on Figure 1 and appears in Appendix C. By combining Lemma 8 with a simple inequality $d_{TV}(X^n, Y^n) \leqslant n\cdot d_{TV}(X,Y)$ (which can be proved by a hybrid argument) we obtain

▶ **Corollary 9.** *Let X, Y be such that (a) $d_{TV}(X;Y) \leqslant \gamma$ and (b) $|H_\alpha(X) - H_\alpha(Y)| \geqslant 2\delta$. Then any (δ,ϵ)-estimator for H_α, where $\epsilon \leqslant \frac{1}{3}$, requires $\frac{1}{3}\gamma^{-1}$ samples.*

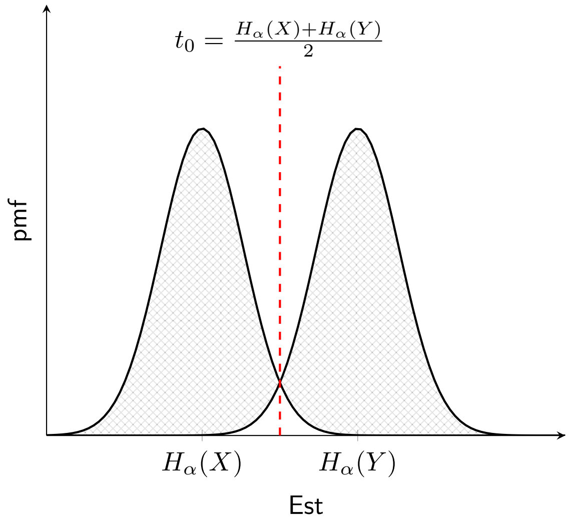


Figure 1 Turning estimators into distinguishers in total variation.

We will need the following inequalities, that refine the known Bernoulli-inequality $(1+u)^\alpha \geqslant 1+\alpha u$ by introducing higher-order terms.

▶ **Proposition 10** (Bernouli-type inequalities). *We have*

$$\forall \alpha > 1,\ \forall u > -1: \quad (1+u)^\alpha \geqslant 1+\alpha u \tag{6}$$

$$\forall \alpha \geqslant 2,\ \forall u > 0: \quad (1+u)^\alpha \geqslant 1+\alpha u + u^\alpha \tag{7}$$

$$\forall \alpha \in [1,2],\ \forall u \in [0,1]: \quad (1+u)^\alpha \geqslant 1+\alpha u + \frac{\alpha(\alpha-1)}{4}u^2 \tag{8}$$

$$\forall \alpha \in [1,2],\ \forall u > 1: \quad (1+u)^\alpha \geqslant 1+\alpha u + \frac{\alpha-1}{3}u^\alpha \tag{9}$$

Proof. To prove Equation (6) consider the function $f(u) = (1+u)^\alpha$. It is convex when $\alpha > 1$, hence its graph is above the tangent line at $u = 0$. This means that $f(u) \geqslant f(0) + \frac{\partial f}{\partial}(0)u$, and since $f(0) = 1$ and $\frac{\partial f}{\partial u}(0) = \alpha$ the inequality follows.

In order to prove Equation (7), we consider the function $f(u) = (1+u)^\alpha - 1 - \alpha u - u^\alpha$. Its derivative equals $\frac{\partial f}{\partial u}(u) = \alpha\left((1+u)^{\alpha-1} - u^{\alpha-1} - 1\right)$. If we show it is non-negative for $u \geqslant 0$, we establish the claimed inequality as then $f(u) \geqslant f(0) \geqslant 0$. We calculate the second derivative $\frac{\partial^2 f}{\partial u^2}(u) = \alpha(\alpha-1)\left((1+u)^{\alpha-2} - u^{\alpha-2}\right)$ and see it is positive when $u \geqslant 0$ (here we use the assumption that $\alpha \geqslant 2$). We conclude that $\frac{\partial f}{\partial u}(u)$ is increasing for $u \geqslant 0$ and hence $\frac{\partial f}{\partial u}(u) \geqslant \frac{\partial f}{\partial u}(0) = 0$, which finishes the proof.

To prove Equation (8) we define $f = (1+u)^\alpha - 1 - \alpha u - \frac{\alpha(\alpha-1)}{4}u^2$. We note that $\frac{\partial f}{\partial u}(u) = \alpha(1+u)^{\alpha-1} - \alpha - \frac{\alpha(\alpha-1)}{2}u$. This function is concave because $\alpha \in [1,2]$. Since $\frac{\partial f}{\partial u}(0) = 0$ and $\frac{\partial f}{\partial u}(1) = \alpha(1+1)^{\alpha-1} - \alpha - \frac{\alpha(\alpha-1)}{2} \geqslant \alpha^2 - \alpha - \frac{\alpha(\alpha-1)}{2} = \frac{1}{2}(\alpha^2-\alpha) \geqslant 0$ (we have used the Bernouli inequality $(1+1)^{\alpha-1} \geqslant 1+\alpha-1$), by concavity we conclude that the $\frac{\partial f}{\partial u}(u) \geqslant 0$ on the whole interval $u \in [0,1]$. This means that f is decreasing and $f(u) \geqslant f(0) = 0$ for $u \in [0,1]$, which establishes the claimed inequality.

[4] We use the fact that α-norms, defined by $\|p\|_\alpha = \left(\sum_i |p_i|^\alpha\right)^{\frac{1}{\alpha}}$, are decreasing in α. The same inequality is applied in [1], the proof of Lemma 2.1.

To obtain Equation (9) we consider the function $f(u) = (1+u)^{\alpha} - 1 - \alpha u - Cu^{\alpha}$. Its derivative equals $\frac{\partial f}{\partial u}(u) = \alpha\left((1+u)^{\alpha-1} - 1 - Cu^{\alpha-1}\right)$. It suffices to choose C such that $f(1) \geqslant 0$ and $\frac{\partial f}{\partial u}(u) \geqslant 0$ for $u \geqslant 1$ as then $f(u) \geqslant 1$ for $u \geqslant 1$. The second derivative equals $\frac{\partial^2 f}{\partial u^2}(u) = \alpha(\alpha-1)\left((1+u)^{\alpha-2} - Cu^{\alpha-2}\right)$, and we conclude that, for $1 \leqslant \alpha \leqslant 2$ and $u \geqslant 1$, it bigger than zero when $C \leqslant 2^{\alpha-2}$. Thus the first derivative increases and is non-negative if, in addition, $\frac{\partial f}{\partial u}(1) \geqslant 0$, that is $C \leqslant 2^{\alpha-1} - 1$. We conclude that $f(u) \geqslant 0$ with $C = \min\left(2^{\alpha-2}, 2^{\alpha-1}-1, 2^{\alpha}-\alpha-1\right)$, that is when $\frac{\partial^2 f}{\partial u^2}(1), \frac{\partial f}{\partial u}(1), f(1)$ are all non-negative. Under the assumption $\alpha \leqslant 2$ this can be simplified to $C = 2^{\alpha} - 1 - \alpha$. We notice further that $2^{\alpha-1} - 1 - \alpha \geqslant (\ln 4 - 1)(\alpha - 1)$ when $\alpha \in (1,2)$ which shows that we can take $C = 0.38(\alpha-1)$. ◀

▶ **Lemma 11** (Distributions with different entropy yet close in total variation). *For any real $\alpha > 1$ and any set S of size $K \geqslant 2$ there exist distributions on S that are γ-close but with Renyi α-entropy different by at least Δ, for any parameters satisfying the following*

- *For any $\Delta \leqslant 1$, any $\alpha \in [1,2]$ and $\gamma = O\left(\max\left(\Delta^{\frac{1}{2}}K^{-\frac{1}{2}}, K^{-1+\frac{1}{\alpha}}\Delta^{\frac{1}{\alpha}}\right)\right)$*
- *For any $\Delta \leqslant 1$, any $\alpha > 2$ and $\gamma = O\left(\Delta^{\frac{1}{\alpha}}K^{-1+\frac{1}{\alpha}}\right)$*
- *For any $\Delta \geqslant 1$, any $\alpha \in [1,2]$ and $\gamma = \max\left(2^{\left(1-\frac{1}{\alpha}\Delta\right)}K^{-1+\frac{1}{\alpha}}, 2^{\frac{1}{2}\Delta}K^{-\frac{1}{2}}\right)$*
- *For any $\Delta \geqslant 1$, any $\alpha > 2$ and $\gamma = O\left(2^{\left(1-\frac{1}{\alpha}\right)\Delta}K^{-1+\frac{1}{\alpha}}\right)$*

In particular, by applying Corollary 9 to the setting in the lemma above, we obtain the lower bounds on the sample complexity.

▶ **Corollary 12** (Estimating entropy with constant additive error). *For any constant $\alpha > 1$, estimating α-entropy with additive error at most 1 requires at least $\Omega(1) \cdot \max\left(K^{\frac{1}{2}}, K^{1-\frac{1}{\alpha}}\right)$ samples. More generally bounds (for any accuracy Δ) apply as shown in Table 1.*

Proof of Lemma 11. Fix a K-element set S and a parameter $\epsilon > 0$ and consider the following pair of distributions (given the choice of X, the choice of Y is close to the "worst" choice as shown in Section D):

(a) X is uniform over S,

(b) Y puts a mass of $\frac{1}{K} + \gamma$ on one fixed point of S and $\frac{1}{K} - \frac{\gamma}{K-1}$ on the remaining points of S,

where the exact value of the parameter γ is to be optimized later. We calculate that

$$\sum_x (P_Y(x))^{\alpha} = \left(K^{-1} + \gamma\right)^{\alpha} + (K-1)\left(K^{-1} - \gamma(K-1)^{-1}\right)^{\alpha}$$

and

$$K^{\alpha} \cdot \sum_x (P_Y(x))^{\alpha} = (1 + K\gamma)^{\alpha} + (K-1)\left(1 - \gamma\frac{K}{K-1}\right)^{\alpha}.$$

Since $\sum_x (P_X(x))^{\alpha} = K^{1-\alpha}$ we get

$$\frac{\sum_x (P_Y(x))^{\alpha}}{\sum_x (P_X(x))^{\alpha}} = K^{-1}\left((1+K\gamma)^{\alpha} + (K-1)\left(1 - \gamma\frac{K}{K-1}\right)^{\alpha}\right). \qquad (10)$$

Now if either $K\gamma \leqslant 1$ and $\alpha \in (1,2)$ or $\alpha \geqslant 2$, by Proposition 10 we obtain

$$(1+K\gamma)^{\alpha} + (K-1)\left(1 - \gamma\frac{K}{K-1}\right)^{\alpha} \geqslant K + \Omega_{\alpha}(1)\min\left((K\gamma)^2, (K\gamma)^{\alpha}\right) \qquad (11)$$

for some constants depending on α, where we have used Equation (6) to lower-bound $\left(1-\gamma\frac{K}{K-1}\right)^{\alpha}$ and Equations (8) and (7) to lower-bound $(1+K\gamma)^{\alpha}$. More precisely, we have

$$(1+K\gamma)^{\alpha}+(K-1)\left(1-\gamma\frac{K}{K-1}\right)^{\alpha} \geqslant \begin{cases} K+\frac{\alpha-1}{3}(K\gamma)^{\alpha} & \text{if } \alpha\in(1,2)\wedge K\gamma>1 \\ K+\frac{\alpha(\alpha-1)}{4}(K\gamma)^{2} & \text{if } \alpha\in(1,2)\wedge K\gamma\leqslant 1 \\ K+(K\gamma)^{\alpha} & \text{if } \alpha>2 \end{cases}$$

Using this bound in the right-hand side of Equation (10), we obtain

$$\left(\frac{\sum_x (P_Y(x))^{\alpha}}{\sum_x (P_X(x))^{\alpha}}\right)^{\frac{1}{\alpha-1}} \geqslant \begin{cases} 1+\frac{\alpha-1}{3}K^{\alpha-1}\gamma^{\alpha} & \text{if } \alpha\in(1,2)\wedge K\gamma>1 \\ 1+\frac{\alpha(\alpha-1)}{4}K\gamma^{2} & \text{if } \alpha\in(1,2)\wedge K\gamma\leqslant 1 \\ 1+K^{\alpha-1}\gamma^{\alpha} & \text{if } \alpha>2 \end{cases} \tag{12}$$

It remains to choose the parameter γ, remembering about the assumptions on γ and α made in Equation (11). We may choose it the following ways:

Case 1: for $\Delta\in(0,1)$ and $\alpha>2$ we will choose: $\frac{1}{\alpha-1}\cdot K^{\alpha-1}\gamma^{\alpha}<1$. By taking the logarithm of Equation (12) and dividing by $\alpha-1$ we obtain

$$H_{\alpha}(Y)-H_{\alpha}(X)\geqslant\frac{1}{\alpha-1}\log\left(1+K^{\alpha-1}\gamma^{\alpha}\right).$$

Now the elementary inequality $\log(1+u)\geqslant u$ valid for $0\leqslant u\leqslant 1$ yields

$$H_{\alpha}(Y)-H_{\alpha}(X)\geqslant\frac{1}{\alpha-1}\cdot K^{\alpha-1}\gamma^{\alpha}.$$

Thus we achieve the entropy gap $\Delta=\frac{1}{\alpha-1}\cdot K^{\alpha-1}\gamma^{\alpha}$ and the distance $\gamma=((\alpha-1)\Delta)^{\frac{1}{\alpha}}K^{-1+\frac{1}{\alpha}}$ for any Δ between 0 and 1.

Case 2: for $\Delta\leqslant 1$ and $\alpha\in(1,2)$ we choose $\min\left(K\gamma^{2},K^{\alpha-1}\gamma^{\alpha}\right)<1$. Using Equation (12), taking the logarithm of both sides and dividing by $\alpha-1$ we obtain

$$H_{\alpha}(Y)-H_{\alpha}(X)>\frac{1}{\alpha-1}\log\left(1+\frac{\alpha(\alpha-1)}{4}\cdot\min\left(K\gamma^{2},K^{\alpha-1}\gamma^{\alpha}\right)\right).$$

Now the elementary inequality $\log(1+u)\geqslant u$ valid for $0\leqslant u\leqslant 1$ yields

$$H_{\alpha}(Y)-H_{\alpha}(X)\geqslant\frac{\alpha}{4}\cdot\min\left(K\gamma^{2},K^{\alpha-1}\gamma^{\alpha}\right).$$

Hence we can have the entropy gap $\Delta=\frac{\alpha}{4}\cdot\min\left(K\gamma^{2},K^{\alpha-1}\gamma^{\alpha}\right)$ and the distance $\gamma=\max\left(K^{-1+\frac{1}{\alpha}}\left(\frac{4\Delta}{\alpha}\right)^{\frac{1}{\alpha}},K^{-\frac{1}{2}}\left(\frac{4\Delta}{\alpha}\right)^{\frac{1}{2}}\right)$. The number Δ can be arbitrary between 0 and 1.

Case 3: for $\Delta>1$ and $\alpha\geqslant 2$ we choose $\frac{1}{\alpha-1}\cdot K^{\alpha-1}\gamma^{\alpha}>1$. Under this assumption, Equation (12) holds with the term $K^{\alpha-1}\gamma^{\alpha}$ on the right-hand side. By taking the logarithm in Equation (12) and dividing by $\alpha-1$ we obtain

$$H_{\alpha}(Y)-H_{\alpha}(X)>\frac{1}{\alpha-1}\cdot\log\left(1+K^{\alpha-1}\gamma^{\alpha}\right).$$

Now the inequality $\log(1+u) > \log u$ implies

$$H_\alpha(Y) - H_\alpha(X) > \frac{1}{\alpha-1}\log\left(K^{\alpha-1}\gamma^\alpha\right).$$

Thus, we can have the entropy gap $\Delta = \frac{1}{\alpha-1}\log\left(K^{\alpha-1}\gamma^\alpha\right)$ and the distance $\gamma = 2^{\Delta\left(1-\frac{1}{\alpha}\right)}K^{-1+\frac{1}{\alpha}}$, for any $1 \leqslant \Delta \leqslant \log K - O(1)$ (the upper bound follows by substituting $\gamma = \frac{K-1}{K}$ which is the maximal value).

Case 4: for $\boldsymbol{\Delta > 1}$ and $\boldsymbol{\alpha \in (1,2)}$ we choose $\boldsymbol{\min\left(K\gamma^2, K^{\alpha-1}\gamma^\alpha\right) > 1}$. Recall, as for Case 2, that for $\alpha < 2$ we have $K^{\alpha-1}\gamma^\alpha > K\gamma^2$ when $K\gamma > 1$. Using this in Equation (12), taking the logarithm of both sides and dividing by $\alpha - 1$ we obtain

$$H_\alpha(Y) - H_\alpha(X) > \frac{1}{\alpha-1}\log\left(1 + \frac{\alpha(\alpha-1)}{4}\cdot\min\left(K\gamma^2, K^{\alpha-1}\gamma^\alpha\right)\right).$$

Now the inequality $\log(1+u) > \log u$ implies

$$H_\alpha(Y) - H_\alpha(X) > \frac{1}{\alpha-1}\log\left(\frac{\alpha(\alpha-1)}{4}\cdot\min\left(K\gamma^2, K^{\alpha-1}\gamma^\alpha\right)\right).$$

Thus, for the entropy gap $\Delta = \frac{1}{\alpha-1}\log\left(\frac{\alpha(\alpha-1)}{4}\cdot\min\left(K\gamma^2, K^{\alpha-1}\gamma^\alpha\right)\right)$ we get the distance $\gamma = \frac{4}{\alpha(\alpha-1)}\cdot\max\left(2^{\Delta\left(1-\frac{1}{\alpha}\right)}K^{-1+\frac{1}{\alpha}}, 2^{\frac{1}{2}\Delta}K^{-\frac{1}{2}}\right)$, for for any $1 \leqslant \Delta \leqslant \frac{1}{\alpha-1}\log K - O(1)$ (the upper bound follows by substituting $\gamma = \frac{K-1}{K}$ which is the maximal value). ◀

6 Conclusion

This paper offers stronger upper and lower bounds on the complexity of estimating Renyi entropy. Except quantitative improvements, it also provides simplifies the analysis, and provides more insight into the technique used to prove lower bounds.

Applying this technique to related problems, e.g. estimating different properties of discrete distributions besides entropy, is an interesting problem for future research.

We also emphasize that our construction for the lower bounds can be somewhat improved in two aspects: firstly, in Lemma 11 the choice of Y is optimal but X may be not - we assumed for simplicity that it is flat; secondly, there may be need for a more carefull bound on the variational distance between n-fold product distributions Lemma 8.

As for upper bounds, it remains an intriguing question if we can obtain improvements also for Shannon entropy estimation in low or medium entropy regimes.

References

1 Jayadev Acharya, Alon Orlitsky, Ananda Theertha Suresh, and Himanshu Tyagi. The complexity of estimating rényi entropy. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1855–1869, 2015. doi:10.1137/1.9781611973730.124.

2 Jayadev Acharya, Alon Orlitsky, Ananda Theertha Suresh, and Himanshu Tyagi. Estimating renyi entropy of discrete distributions. *IEEE Trans. Information Theory*, 63(1):38–56, 2017. doi:10.1109/TIT.2016.2620435.

3 Erdal Arikan. An inequality on guessing and its application to sequential decoding. *IEEE Trans. Information Theory*, 42(1):99–105, 1996. doi:10.1109/18.481781.

4 Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, François-Xavier Standaert, and Yu Yu. Leftover hash lemma, revisited. In *Advances in Cryptology – CRYPTO 2011 – 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, pages 1–20, 2011. doi:10.1007/978-3-642-22792-9_1.

5 Tugkan Batu, Sanjoy Dasgupta, Ravi Kumar, and Ronitt Rubinfeld. The complexity of approximating entropy. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 678–687, 2002. doi:10.1145/509907.510005.

6 Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren D. Smith, and Patrick White. Testing closeness of discrete distributions. *J. ACM*, 60(1):4:1–4:25, 2013. doi:10.1145/2432622.2432626.

7 Charles H. Bennett, Gilles Brassard, Claude Crépeau, and Ueli M. Maurer. Generalized privacy amplification. *IEEE Trans. Information Theory*, 41(6):1915–1923, 1995. doi:10.1109/18.476316.

8 Yevgeniy Dodis and Yu Yu. Overcoming weak expectations. In *Theory of Cryptography – 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, pages 1–22, 2013. doi:10.1007/978-3-642-36594-2_1.

9 Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation – In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, pages 68–75. 2011. doi:10.1007/978-3-642-22670-0_9.

10 Manjesh Kumar Hanawal and Rajesh Sundaresan. Guessing revisited: A large deviations approach. *IEEE Trans. Information Theory*, 57(1):70–78, 2011. doi:10.1109/TIT.2010.2090221.

11 Russell Impagliazzo and David Zuckerman. How to recycle random bits. In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October – 1 November 1989*, pages 248–253, 1989. doi:10.1109/SFCS.1989.63486.

12 R. Jenssen, K. E. Hild, D. Erdogmus, J. C. Principe, and T. Eltoft. Clustering using renyi's entropy. In *Proceedings of the International Joint Conference on Neural Networks, 2003.*, volume 1, pages 523–528 vol.1, July 2003. doi:10.1109/IJCNN.2003.1223401.

13 Petr Jizba, Hagen Kleinert, and Mohammad Shefaat. Rényi's information transfer between financial time series. *Physica A: Statistical Mechanics and its Applications*, 391(10):2971–2989, 2012. doi:10.1016/j.physa.2011.12.064.

14 Donald E. Knuth. *The Art of Computer Programming, Volume 3: (2nd Ed.) Sorting and Searching.* Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1998.

15 Ke Li, Wanlei Zhou, Shui Yu, and Bo Dai. Effective ddos attacks detection using generalized entropy metric. In *Algorithms and Architectures for Parallel Processing, 9th International Conference, ICA3PP 2009, Taipei, Taiwan, June 8-11, 2009. Proceedings*, pages 266–280, 2009. doi:10.1007/978-3-642-03095-6_27.

16 Bing Ma, Alfred O. Hero III, John D. Gorman, and Olivier J. J. Michel. Image registration with minimum spanning tree algorithm. In *Proceedings of the 2000 International Conference on Image Processing, ICIP 2000, Vancouver, BC, Canada, September 10-13, 2000*, pages 481–484, 2000. doi:10.1109/ICIP.2000.901000.

17 Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Multiple source adaptation and the rényi divergence. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, pages 367–374,

2009. URL: https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1600&proceeding_id=25.

18 Albert W. Marshall, Ingram Olkin, and Barry C. Arnold. *Inequalities : Theory of Majorization and its Applications*. Springer Science+Business Media, LLC, New York, 2011.

19 Abolfazl S. Motahari, Guy Bresler, and David N. C. Tse. Information theory of DNA shotgun sequencing. *IEEE Trans. Information Theory*, 59(10):6273–6289, 2013. doi:10.1109/TIT.2013.2270273.

20 Huzefa Neemuchwala, Alfred O. Hero III, Sakina Zabuawala, and Paul L. Carson. Image registration methods in high-dimensional space. *Int. J. Imaging Systems and Technology*, 16(5):130–145, 2006. doi:10.1002/ima.20079.

21 Dávid Pál, Barnabás Póczos, and Csaba Szepesvári. Estimation of rényi entropy and mutual information based on generalized nearest-neighbor graphs. In *Proceedings of the 23rd International Conference on Neural Information Processing Systems*, NIPS'10, pages 1849–1857, USA, 2010. Curran Associates Inc. URL: http://dl.acm.org/citation.cfm?id=2997046.2997102.

22 Liam Paninski. Estimation of entropy and mutual information. *Neural Comput.*, 15(6):1191–1253, June 2003. doi:10.1162/089976603321780272.

23 Liam Paninski. A coincidence-based test for uniformity given very sparsely sampled discrete data. *IEEE Trans. Information Theory*, 54(10):4750–4755, 2008. doi:10.1109/TIT.2008.928987.

24 C. E. Pfister and W. G. Sullivan. Rényi entropy, guesswork moments, and large deviations. *IEEE Trans. Information Theory*, 50(11):2794–2800, 2004. doi:10.1109/TIT.2004.836665.

25 A. Renyi. On measures of information and entropy. In *Proceedings of the 4th Berkeley Symposium on Mathematics, Statistics and Probability*, pages 547–561, 1960. URL: http://digitalassets.lib.berkeley.edu/math/ucb/text/math_s4_v1_article-27.pdf.

26 Prasanna K. Sahoo and Gurdial Arora. A thresholding method based on two-dimensional renyi's entropy. *Pattern Recognition*, 37(6):1149–1161, 2004. doi:10.1016/j.patcog.2003.10.008.

27 C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, January 2001. doi:10.1145/584091.584093.

28 Gregory Valiant and Paul Valiant. Estimating the unseen: an n/log(n)-sample estimator for entropy and support size, shown optimal via new clts. In *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 685–694, 2011. doi:10.1145/1993636.1993727.

29 Paul C. van Oorschot and Michael J. Wiener. Parallel collision search with cryptanalytic applications. *J. Cryptology*, 12(1):1–28, 1999. doi:10.1007/PL00003816.

30 Dongxin Xu. *Energy, Entropy and Information Potential for Neural Computation*. PhD thesis, University of Florida, Gainesville, FL, USA, 1998. AAI9935317.

31 Dongxin Xu and Deniz Erdogmuns. *Renyi's Entropy, Divergence and Their Nonparametric Estimators*, pages 47–102. Springer New York, New York, NY, 2010. doi:10.1007/978-1-4419-1570-2_2.

A Proof of Lemma 4

Proof. The proof of Equation (2) goes by induction. It is clearly valid for $\alpha = 1$. Assuming that it is valid for some $\alpha \geqslant 1$, we obtain

$$
\begin{aligned}
n(x)^{\underline{\alpha+1}} &= n(x)^{\underline{\alpha}} \cdot (n(x)-\alpha) \\
&= \sum_{i_1\neq i_2\neq\ldots\neq i_\alpha} \xi_{i_1}(x)\xi_{i_2}(x)\cdot\ldots\cdot\xi_{i_\alpha}(x)\cdot\sum_{i_{\alpha+1}}(\xi_{i_{\alpha+1}}(x)-\alpha) \\
&= -\alpha \sum_{i_1\neq i_2\neq\ldots\neq i_\alpha} \xi_{i_1}(x)\xi_{i_2}(x)\cdot\ldots\cdot\xi_{i_\alpha}(x)+ \\
&\quad + \sum_{i_1\neq i_2\neq\ldots\neq i_\alpha\neq i_{\alpha+1}} \xi_{i_1}(x)\xi_{i_2}(x)\cdot\ldots\cdot\xi_{i_\alpha}(x) \\
&\quad + \sum_{\substack{i_1\neq i_2\neq\ldots\neq i_\alpha \\ i_{\alpha+1}\in\{i_1,\ldots,i_\alpha\}}} \xi_{i_1}(x)\xi_{i_2}(x)\cdot\ldots\cdot\xi_{i_\alpha}(x)\xi_{i_{\alpha+1}}(x).
\end{aligned}
$$

Since ξ_i are boolean we have

$$
\sum_{\substack{i_1\neq i_2\neq\ldots\neq i_\alpha \\ i_{\alpha+1}\in\{i_1,\ldots,i_\alpha\}}} \xi_{i_1}(x)\xi_{i_2}(x)\cdot\ldots\cdot\xi_{i_\alpha}(x)\xi_{i_{\alpha+1}}(x) = \\
\alpha\cdot\sum_{i_1\neq i_2\neq\ldots\neq i_\alpha} \xi_{i_1}(x)\xi_{i_2}(x)\cdot\ldots\cdot\xi_{i_\alpha}(x)
$$

By putting together the last two equations we end the proof of Equation (2). To get Equation (3) we simply take the expectation and use independence. ◀

B Proof of Lemma 5

Proof. Note that

$$
\begin{aligned}
\left(\sum_x n(x)^{\underline{\alpha}}\right)^2 &= \sum_{x,y}\sum_{\substack{i_1\neq i_2\neq\ldots\neq i_\alpha \\ j_1\neq j_2\neq\ldots\neq j_\alpha}}\prod_{r=1}^{\alpha}\xi_{i_r}(x)\xi_{j_r}(y) \\
&= \sum_{x\neq y}\sum_{i_1\neq i_2\neq\ldots\neq i_\alpha\neq j_1\neq j_2\neq\ldots\neq j_\alpha}\prod_{r=1}^{\alpha}\xi_{i_r}(x)\xi_{j_r}(y)+ \\
&\quad +\sum_{x}\sum_{\substack{i_1\neq i_2\neq\ldots\neq i_\alpha \\ j_1\neq j_2\neq\ldots\neq j_\alpha}}\prod_{r=1}^{\alpha}\xi_{i_r}(x)\xi_{j_r}(x).
\end{aligned}
$$

Now we have

$$
\begin{aligned}
I_1 &= \mathbb{E}\left[\sum_{x\neq y}\sum_{i_1\neq i_2\neq\ldots\neq i_\alpha\neq j_1\neq j_2\neq\ldots\neq j_\alpha}\prod_{r=1}^{\alpha}\xi_{i_r}(x)\xi_{j_r}(y)\right] \\
&= n^{\underline{2\alpha}}\sum_{x\neq y}p(x)^\alpha p(y)^\alpha \\
&= n^{\underline{2\alpha}}\left((p_\alpha)^2 - p_{2\alpha}\right).
\end{aligned}
$$

Also

$$
\begin{aligned}
I_2 &= \mathbb{E}\left[\sum_{x}\sum_{\substack{i_1\neq i_2\neq\ldots\neq i_\alpha\\ j_1\neq j_2\neq\ldots\neq j_\alpha}}\prod_{r=1}^{\alpha}\xi_{i_r}(x)\xi_{j_r}(x)\right]\\
&= \mathbb{E}\left[\sum_{x\in\mathcal{X}}\sum_{\ell=0}^{\alpha}\sum_{\substack{i_1\neq i_2\neq\ldots\neq i_\alpha\\ j_1\neq j_2\neq\ldots\neq j_\alpha\\ |\{i_1\neq i_2\neq\ldots\neq i_\alpha\}\cap\{j_1\neq j_2\neq\ldots\neq j_\alpha\}|=\ell}}\prod_{r=1}^{\alpha}\xi_{i_r}(x)\xi_{j_r}(x)\right]\\
&= \sum_{x\in\mathcal{X}}\sum_{\ell=0}^{\alpha} n^{\underline{\alpha}}(n-\alpha)^{\underline{\alpha-\ell}}\binom{\alpha}{\ell}^2 l!\cdot p(x)^{2\alpha-\ell}\\
&= \sum_{\ell=0}^{\alpha} n^{\underline{\alpha}}(n-\alpha)^{\underline{\alpha-\ell}}\binom{\alpha}{\ell}^2 l!\cdot p_{2\alpha-\ell}\\
&= n^{\underline{2\alpha}}p_{2\alpha} + \sum_{\ell=1}^{\alpha} n^{\underline{\alpha}}(n-\alpha)^{\underline{\alpha-\ell}}\binom{\alpha}{\ell}^2 l!\cdot p_{2\alpha-\ell},
\end{aligned}
$$

where we observed that if the sets $\{i_1,\ldots,i_\alpha\}$ and $\{j_1,\ldots,j_\alpha\}$ have exactly ℓ common elements then $\mathbb{E}\prod_{r=1}^{\alpha}\xi_{i_r}(x)\xi_{j_r}(x) = p(x)^{2\alpha-\ell}$, and that there are $n^{\underline{\alpha}}(n-\alpha)^{\underline{\alpha-\ell}}\binom{\alpha}{\ell}^2 l!$ choices for the such sets $\{i_1,\ldots,i_\alpha\}$ and $\{j_1,\ldots,j_\alpha\}$[5]. Putting this all together we obtain

$$
\begin{aligned}
\operatorname{Var}\left[\sum_{x} n(x)^{\underline{\alpha}}\right] &= n^{\underline{2\alpha}}(p_\alpha)^2 + \sum_{\ell=1}^{\alpha} n^{\underline{\alpha}}(n-\alpha)^{\underline{\alpha-\ell}}\binom{\alpha}{\ell}^2 l!\cdot p_{2\alpha-\ell} - (n^{\underline{\alpha}}p_\alpha)^2\\
&= n^{\underline{\alpha}}((n-\alpha)^{\underline{\alpha}} - n^{\underline{\alpha}})(p_\alpha)^2 + \sum_{\ell=1}^{\alpha} n^{\underline{\alpha}}(n-\alpha)^{\underline{\alpha-\ell}}\binom{\alpha}{\ell}^2 l!\cdot p_{2\alpha-\ell}
\end{aligned}
$$

which completes the proof. ◀

C Proof of Lemma 8

Proof. We will use the fact that if two distributions are ϵ-close (i.e. $d_{TV}(X',Y') < \epsilon$) then no distinguisher can distinguish between them with advantage greater then $\frac{\epsilon}{2}$. Let us assume that $|H_\alpha(X) - H_\alpha(Y)| \geqslant 2\delta$, then by using estimator $\hat{f}$ as part of the distinguisher i.e. if $|\hat{f}(.) - H_\alpha(X)| \leq \delta$ then distinguisher "guesses" that initial distribution was X^n, else "guesses" Y^n. Now we notice that initial distribution was X^n distinguisher will "guess" correctly with probability $1-\epsilon$, and if the initial distribution was Y^n then estimator with probability $1-\epsilon$ will output value in $[H_\alpha(Y)-\delta\ ;\ H_\alpha(Y)+\delta]$ thus distinguisher will guess correctly again. Our distinguisher achieves $1/2-\epsilon$ advantage thus we deduce that $d_{TV}(X^n;Y^n) > 1-2\epsilon$. ◀

[5] For a quick sanity check of this formula, note that when $p_i = 1$ (a constant random variable) then we should get $(n^{\underline{\alpha}})^2 = \sum_{\ell=0}^{\alpha} n^{\underline{\alpha}}(n-\alpha)^{\underline{\alpha-\ell}}\binom{\alpha}{\ell}^2 l!$. For $\alpha = 2$ this reduces to the identity $n(n-1) = (n-2)(n-3) + 4(n-2) + 2$.

D Maximizing entropy gap within variational distance constraints

▶ **Theorem 13.** *Let q be a fixed distribution over k elements, and $\alpha > 1$, $\epsilon \in (0,1)$ be fixed. Suppose that $q_1 \geqslant q_2 \geqslant \ldots \geqslant q_k$. Then the distribution p which is ϵ-close to q and has minimal possible α-entropy is given by*

$$q_i = \begin{cases} p_1 + \epsilon & i = 1 \\ p_i & 1 < i < i_0 \\ p_{i_0} - \sum_{j \geqslant i_0} p_j & i = i_0 \\ 0 & i > i_0 \end{cases} \tag{13}$$

where i_0 is the biggest number such that $\sum_{i \geqslant i_0} p_i \geqslant \epsilon$, for some x_0 such that $p(x_0)$ is the biggest mass, and for some $\epsilon' < \epsilon$.

Proof. We will apply majorization techniques [18]. Let q be optimal. Suppose that $q(x_1) > p(x_1)$ and $q(x_2) > p(x_2)$ where $x_1 \neq x_2$. Since q has the biggest possible power sum $S(q) = \sum_x q(x)^\alpha$ we see that $p(x_1)$ and $p(x_2)$ are two biggest probability masses. Assume, without loss of generality, that $q(x_1) \geqslant q(x_2)$. For some small $\delta > 0$ we perturb q into q' such that $q'(x_1) = q(x_1) + \delta$ and $q'(x_1) = q(x_1) - \delta$ and $q'(x) = q(x)$. Note that for small δ the distance between q' and p is at most as between p and q, and that q' majorizes q (considered as vectors) and the power sum $S(q)$ is Schur convex, hence $S(q) > S(q')$. The contradiction means that $q(x) > p(x)$ for only one $x = x_0$.

Consider now the smallest values $q(x_1), q(x_2)$ such that $0 < q(x_1) < p(x_1), 0 < q(x_2) < p(x_2)$ for $x_1 \neq x_2$ that are strictly bigger than zero. For some small $\delta > 0$ we perturb q into q' such that $q'(x_1) = q(x_1) + \delta$ and $q'(x_1) = q(x_1) - \delta$ and $q'(x) = q(x)$. We see that for δ small enough the distance from q' to p is at most as from q to p and that q' majorizes q which means $S(q') > S(q)$. The contradiction means that $0 < q(x) < p(x)$ for at most one $x = x_0$. ◀

Approximating Sparsest Cut in Low Rank Graphs via Embeddings from Approximately Low Dimensional Spaces

Yuval Rabani[1] **and Rakesh Venkat**[*2]

1 **Hebrew University of Jerusalem, Jerusalem, Israel**
yrabani@cs.huji.ac.il

2 **Hebrew University of Jerusalem, Jerusalem, Israel**
rakesh@cs.huji.ac.il

Abstract

We consider the problem of embedding a finite set of points $\{x_1, \ldots, x_n\} \in \mathbb{R}^d$ that satisfy ℓ_2^2 triangle inequalities into ℓ_1, when the points are *approximately* low-dimensional. Goemans (unpublished, appears in [20]) showed that such points residing in *exactly* d dimensions can be embedded into ℓ_1 with distortion at most $\sqrt{d}$. We prove the following robust analogue of this statement: if there exists a r-dimensional subspace Π such that the projections onto this subspace satisfy $\sum_{i,j\in[n]} \|\Pi x_i - \Pi x_j\|_2^2 \geq \Omega(1) \sum_{i,j\in[n]} \|x_i - x_j\|_2^2$, then there is an embedding of the points into ℓ_1 with $O(\sqrt{r})$ average distortion. A consequence of this result is that the integrality gap of the well-known Goemans-Linial SDP relaxation for the Uniform Sparsest Cut problem is $O(\sqrt{r})$ on graphs G whose r-th smallest normalized eigenvalue of the Laplacian satisfies $\lambda_r(G)/n \geq \Omega(1)\Phi_{SDP}(G)$. Our result improves upon the previously known bound of $O(r)$ on the average distortion, and the integrality gap of the Goemans-Linial SDP under the same preconditions, proven in [7, 6].

1998 ACM Subject Classification F.2.2 Non-Numerical Algorithms

Keywords and phrases Metric Embeddings, Sparsest Cut, Negative type metrics, Approximation Algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.21

1 Introduction

A finite metric space consists of a pair $(\mathcal{X}, d)$, where $\mathcal{X}$ is a finite set of points, and $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ is a distance function on pairs of points in $\mathcal{X}$. Many combinatorial optimization problems can be naturally formulated as a maximization or minimization problem over metric spaces $(\mathcal{X}, d)$ of some target class. However, since it might be computationally difficult to optimize over this class, one considers a *relaxation* that finds a solution $(\mathcal{Y}, d')$ amongst a class of computationally 'easy' metrics, and then looks to produce an *embedding* $\mathcal{Y} \hookrightarrow \mathcal{X}$ into the target space, while minimizing some measure of *distortion* between the distance functions d and d' incurred by the embedding. There has been much work that investigates various measures and costs of distortion incurred by embeddings between metric spaces, and applications thereof (see the surveys [12, 21, 18] and references therein).

In this work, we look at embeddings from ℓ_2^2 metrics to ℓ_1 metrics, motivated by applications to the Sparsest Cut problem. A ℓ_1 metric (or a ℓ_1 space) consists of a finite set of

[*] Research supported by an I-CORE Algorithms Fellowship.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 21; pp. 21:1–21:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

points represented in $\mathbb{R}^d$ with the distance given by the ℓ_1 distance between them. It is a natural target space that can be viewed as an non-negative combination of 'cut-metrics' on the underlying point set, and hence arises frequently in graph-cut based problems. A ℓ_2^2 space, on the other hand, is easy to optimize over, and consists of a finite set of points, say $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$, that satisfy triangle inequalities on the *squares* of distances:

$$\|x_i - x_j\|_2^2 + \|x_j - x_k\|_2^2 \geq \|x_i - x_k\|_2^2 \qquad \forall\ i, j, k \in [n]. \tag{1.1}$$

The Sparsest Cut problem is a fundamental NP-hard graph optimization problem that serves as a striking example of the utility of the metric embedding approach. In the (Uniform) Sparsest Cut problem, we are given a graph $G = (V, c)$, with a symmetric weight function c_{ij} on pairs $\{i, j\}$. The goal is to find a cut $(S, \overline{S})$ of minimum *sparsity* $\Phi(S)$, defined as follows (here, $\mathbb{I}_S(i)$ is 1, if $i \in S$, and 0 otherwise).

$$\Phi(S) := \frac{\sum_{i<j} c_{ij} \left|\mathbb{I}_S(i) - \mathbb{I}_S(j)\right|}{\sum_{i<j} \left|\mathbb{I}_S(i) - \mathbb{I}_S(j)\right|}\,.$$

The best known approximation for the Sparsest Cut problem is due to Arora, Rao and Vazirani [3] (henceforth called the ARV algorithm), who considered the following semidefinite programming relaxation (SDP) introduced by Goemans and Linial (see [9] and [18]).

$$\textbf{SDP-1:}\quad \Phi_{SDP}(G) := \min_{\{x_i\}_{i\in[n]}} \frac{1}{n^2}\sum_{ij} c_{ij}\,\|x_i - x_j\|_2^2$$

$$\text{s.t}\quad \begin{cases} \|x_i - x_j\|_2^2 + \|x_j - x_k\|_2^2 \geq \|x_i - x_k\|_2^2 & \forall i, j, k \in [n]. \\ \sum_{kl} \|x_k - x_l\|_2^2 = n^2. \end{cases}$$

Clearly, $\Phi_{SDP}(G) \leq \Phi(G)$. Notice that any feasible solution to the above SDP constitutes a ℓ_2^2 space. The ARV algorithm works by producing an embedding of the solutions of the above SDP into a ℓ_1 space, with *average distortion* (see Section 2 for a definition) $O(\sqrt{\log n})$. It was shown in [19, 4] that producing an embedding of the SDP solutions into a ℓ_1 space with average distortion D suffices to get a $O(D)$ approximation to the Uniform Sparsest Cut problem.

Though the solutions to SDP-1 can lie in up to n dimensions, for certain graph classes, they are more structured. In particular, if the r-th smallest eigenvalue of the graph Laplacian satisfies $\lambda_r(G)/n \gg \Phi_{SDP}(G)$, then it turns out that the solutions are *approximately* r-dimensional (see Definition 1.2 and Section 3.4). Graphs whose r-th smallest eigenvalue is bounded away from 0 for a typically small r are called *low threshold-rank* graphs; note that spectral expanders are a special case of these for $r = 2$. The work of Guruswami and Sinop [11] exploited higher levels of the Lasserre SDP hierarchy [16], along with the above structure, to give constant-factor guarantees for Sparsest Cut on these graphs. However, this involved partially solving a SDP of size $n^{O(r)}$[1], and did not say anything about the behaviour of the Goemans-Linial SDP on these graphs.

Goemans showed that if the points satisfying ℓ_2^2 triangle inequalities lie in d dimensions, then they can be embedded into ℓ_2 (and hence into ℓ_1, since there is an isometry from ℓ_2 to ℓ_1 [21]) with $\sqrt{d}$ distortion (unpublished, appears in [20], see also [6, Section 4] for an alternative proof).

[1] In a separate work, Guruswami and Sinop [10] give an algorithm that solves the SDP partially, running in $2^{O(r)}\text{poly}(n)$ time, and suffices for their algorithm.

▶ **Theorem 1.1** (Goemans [20, Appendix B]). *Let $x_1, x_2, \dots, x_n \in \mathbb{R}^d$ be n points satisfying ℓ_2^2 triangle inequalities. Then there exists an embedding of these points into ℓ_2, $x_i \mapsto f(x_i)$, with distortion $\sqrt{d}$, that is,*

$$\frac{1}{\sqrt{d}} \ \|x_i - x_j\|_2^2 \le \|f(x_i) - f(x_j)\|_2 \le \|x_i - x_j\|_2^2 , \quad \forall\ i, j \in V.$$

The immediate question that this raises is the following: can one reduce the dimension of ℓ_2^2 metrics, while preserving pairwise distances, *and* the ℓ_2^2 triangle inequalities? The Johnson-Lindenstrauss lemma [13] reduces the dimension to $O(\log n)$, while preserving pairwise distances approximately. However, this procedure does not preserve the ℓ_2^2 triangle inequalities, if the original points satisfied them. In fact, Magen and Moharammi [20] prove a strong lower bound against dimension reduction for ℓ_2^2 metrics.

It is interesting to note that the Johnson-Lindenstrauss lemma, while not preserving the ℓ_2^2 triangle inequalities exactly, does preserve them *approximately*, that is, every sequence of $k \le n$ points $x_{i_1}, \dots, x_{i_k}$ satisfies $\sum_{j=1}^{k-1} \left\|x_{i_j} - x_{i_{j+1}}\right\|_2^2 \ge \beta \cdot \|x_{i_1} - x_{i_k}\|_2^2$, for some $\beta = \Omega(1)$. An observation by Luca Trevisan (personal communication) shows that, in fact, Goemans' theorem is also true for points satisfying approximate triangle inequalities, and the proof uses the ARV algorithm and analysis. However, even this does not yield anything better than $O(\sqrt{\log n})$ for approximately r-dimensional points, when r is small.

The above discussion motivates one to ask if there is a more 'robust' analogue of Goemans' theorem that can be applied to low threshold-rank graphs. Deshpande, Harsha and Venkat [6] considered this question, and showed that one can prove a similar theorem for the case where the points are in approximately r dimensions, albeit giving a bound of $O(r)$ on the *average* distortion (which suffices for Sparsest Cut). One would expect an exact analogue to have a bound of $O(\sqrt{r})$, and it was left open if one could find such an embedding.

We show that there is, indeed, an embedding into ℓ_1 (in fact, into ℓ_2, since all our embeddings are one-dimensional) with $O(\sqrt{r})$ average distortion when the points are approximately r-dimensional.

1.1 Our Results

In order to state our main result, we use the following definition to quantify the notion of approximate rank of a set of points:

▶ **Definition 1.2.** (η-Subspace rank) For any $\eta \in (0, 1]$, a set of points $X = \{x_1, \dots, x_n\} \subseteq \mathbb{R}^d$ will be said to have η-subspace rank r, denoted by $\mathrm{ssr}_\eta(X) = r$, if there exists a subspace given by a projector $\Pi \in \mathbb{R}^{d \times d}$ with rank $(\Pi) = r$ that satisfies:

$$\sum_{i,j \in [n]} \|\Pi x_i - \Pi x_j\|_2^2 \ \ge \ \eta \sum_{i,j \in [n]} \|x_i - x_j\|_2^2 . \tag{1.2}$$

In this work, we will always consider $\eta = \Omega(1)$.

▶ Remark. Since the subspace Π_r defined by the top-r left singular vectors of the matrix M with columns $\{x_i - x_j\}_{ij}$ satisfies $\|\Pi_r M\|_F^2 \ge \left\|\widetilde{\Pi} M\right\|_F^2$ for every $\widetilde{\Pi}$ with rank $\left(\widetilde{\Pi}\right) \le r$, we can always assume that $\Pi = \Pi_r(M)$ when we need to explicitly use the projections. Also, note that the subspace rank is independent of any scaling or shifting of the points, and is always at most the rank of the point set.

Deshpande et al. [6] use a slightly different notion of approximate dimension, called the *stable-rank* of the point set, defined as $\mathrm{sr}\,(M) = \|M\|_F^2 / \sigma_1(M)^2$, where σ_1 is the maximum singular value of the matrix M. Clearly, $\mathrm{sr}\,(M) \le \mathrm{ssr}_\eta(X)/\eta$, and so points with low subspace rank also have low stable rank. While the stable rank is a well-known proxy for rank (see [5, 25]), for applications to the Sparsest Cut problem, the notion of subspace rank suffices and is natural (see Section 3.4). It would be interesting to see if other notions of approximate rank yield further applications or improvements, in Sparsest Cut, or elsewhere.

Our main result is the following:

▶ **Theorem 1.3.** *Given a set of points $X = \{x_1, \dots, x_n\} \in \mathbb{R}^d$ with $\mathrm{ssr}_\eta(X) = r$ that satisfy the ℓ_2^2 triangle inequalities, there is an embedding $X \hookrightarrow \ell_1$ with average distortion at most $O_\eta(\sqrt{r})$. That is, there is a constant $c(\eta)$ and a mapping $h : X \to \mathbb{R}^{d'}$ that satisfies:*

$$\|h(x_i) - h(x_j)\|_1 \le \|x_i - x_j\|_2^2 \qquad \forall i, j \in [n] \tag{1.3}$$

$$\sum_{i,j\in[n]} \|h(x_i) - h(x_j)\|_1 \ge \frac{c(\eta)}{\sqrt{r}} \cdot \sum_{ij} \|x_i - x_j\|_2^2 \tag{1.4}$$

This matches Goemans' theorem in terms of the dependence on r, albeit for average-case distortion. Since the subspace rank is an *average* global condition on the point set, we cannot hope to prove a worst-case distortion guarantee like Goemans' theorem that depends only on the subspace rank (see Appendix A.1).

The above theorem holds even if the points satisfy the ℓ_2^2 triangle inequalities only *approximately*, since the steps in the analysis of the algorithm only need the points to satisfy the approximate version of the triangle inequalities (recall the remarks following Theorem 1.1). Improving on the $\sqrt{r}$ bound above with any technique that works with approximate triangle inequalities would imply an improvement over the ARV algorithm's guarantee, since dimension reduction using the Johnson-Lindenstrauss [13] transform preserves pairwise distances (and hence the ℓ_2^2 inequalities) approximately, while reducing the dimension to $O(\log n)$. Note that this, thus, recovers the unconditional guarantee of $O(\sqrt{\log n})$ of the ARV algorithm, but gives better results for points in lower approximate dimension. This is unsurprising, since our techniques do build on the ARV analysis.

Our main result immediately implies a $O(\sqrt{r})$ approximation algorithm for the Uniform Sparsest Cut problem on low threshold-rank graphs, using just the Goemans-Linial SDP.

▶ **Corollary 1.4.** *Let $\epsilon \in (0, 1]$. Given a regular graph G with r-th smallest eigenvalue of the normalized Laplacian satisfying $\lambda_r(G) \ge \Phi_{SDP}(G)/(1 - \epsilon)$, we can find a $O_\epsilon(\sqrt{r})$ approximation to the sparsest cut in the graph using SDP-1.*

This improves upon the previously known guarantee of $O(r/\epsilon)$ using the Goemans-Linial SDP in [6], under the same precondition.

Proof Techniques

In order to prove our main result, we follow the generic approach of the ARV algorithm [3] that proceeds in two steps: If there is a dense cluster of the solution vectors, then a specific Fréchet embedding (see Section 2 for a definition) works. If not, then the solutions are 'well-spread', and one can always find two $\Omega(n)$-sized sets that are $O(1/\sqrt{\log n})$-apart in ℓ_2^2 distance, using a separating hyperplane algorithm. This constitutes the core of the proof, and the analysis involves a 'chaining argument' which relies on the concentration of measure in high-dimensional spaces. These well-separated sets can then be used to construct a good Fréchet embedding into ℓ_1.

In our case, we would analogously like to find two large sets that are $\Omega(1/\sqrt{r})$-apart, and to do this, we need to work with the *projections* of the points. Note that the projections need not be in ℓ_2^2, while the ARV algorithm's analysis requires the use of ℓ_2^2 triangle inequalities at various points.

Thus, in order to prove Theorem 1.3, we follow and adapt the techniques in Naor, Rabani and Sinclair [22] (henceforth called the NRS analysis). Their work generalized the ARV algorithm's analysis to apply to the more general case of metrics *quasisymmetrically embeddable* into ℓ_2, which includes ℓ_2^2 as a special case. We do not need the complete machinery developed by them, though, and extend only a part of their analysis to our setting. In particular, the chaining argument in [22] works in Euclidean, rather than ℓ_2^2 space, making it useful in our case.

Our result, thus, also demonstrates the utility of isolating the chaining argument from the use of ℓ_2^2 triangle inequalities in the ARV algorithm's analysis.

1.2 Other related Work

We recall that the best known upper bound for the worst-case distortion of embedding $\ell_2^2 \hookrightarrow \ell_1$ is $O(\sqrt{\log n} \cdot \log\log n)$ by [2], building on the techniques in [3, 17]. The best known lower bound is $\Omega(\sqrt{\log n})$ for worst-case distortion [23], and $\exp(\Omega(\sqrt{\log\log n}))$ for average distortion [14]. On *low threshold-rank* graphs (where $\lambda_r \geq \Omega(1)\Phi_{SDP}$), an approximation guarantee of $O(1)$ for Sparsest Cut was obtained using $O(r)$ levels of the Lasserre hierarchy for SDPs [11]. In contrast, the works [7, 6] obtained a weaker $O(r)$ approximation, but using just the basic SDP relaxation. Oveis Gharan and Trevisan [8] also give a rounding algorithm for the basic SDP relaxation on low-threshold rank graphs, but require a stricter pre-condition on the eigenvalues ($\lambda_r \gg \log^{2.5} r \cdot \Phi(G)$), and leverage it to give a stronger $O(\sqrt{\log r})$-approximation guarantee. Their improvement comes from a new structure theorem on the SDP solutions of low threshold-rank graphs being clustered, and using the techniques in ARV for analysis.

Kwok et al. [15] showed that a better analysis of Cheeger's inequality gives a $O(r \cdot \sqrt{1/\lambda_r})$ approximation to the sparsest cut on regular graphs. In particular, when $\lambda_r(G) \geq \epsilon$, this gives a $O(r/\sqrt{\epsilon})$ approximation. Note that our result gives a better approximation in this setting (see Section 3.4).

2 Notation

We use $[n] = \{1, \ldots, n\}$. For a matrix $M \in \mathbb{R}^{d\times d}$, we say $M \succeq 0$ or M is positive-semidefinite (psd) if $y^T X y \geq 0$ for all $y \in \mathbb{R}^d$. The unit Euclidean Ball in $\mathbb{R}^d$ is denoted by B_2^d.

Graphs and Laplacians: All graphs will be defined on a vertex set $V = [n]$ of size n. The vertices will usually be referred to by indices $i, j, k, l \in [n]$. Given a graph with a symmetric weight function on pairs $W : V \times V \mapsto \mathbb{R}^+$, with $W(i,i) = 0\,\forall i$, let $D(i) := \sum_j W(i,j)$ be the degree of vertex $i \in V$. The (normalized) graph Laplacian matrix is defined as:

$$L_W(i,j) := \begin{cases} -\frac{W(i,j)}{\sqrt{D(i)D(j)}} & \text{if } i \neq j\,, \\ 1 & \text{if } i = j\,. \end{cases}$$

Note that $L_W \succeq 0$. We will denote the eigenvalues of (the Laplacian of) the graph G by $0 = \lambda_1(G) \leq \lambda_2(G) \ldots \leq \lambda_n(G)$, in *increasing* order. If the graph is c-*regular*, we have $D(i) = c$ for every $i \in V$. Note that c might be a fraction.

For nodes i, j in G, $d_G(i, j)$ is the shortest path between vertices i, j in G. For $S \subseteq [n]$, $G[S]$ is the subgraph induced by G on S. The *vertex expansion* of G, denoted by $h(G)$ is defined as the largest constant h such that for every set $S \subseteq V$ with $1 \geq |S| \geq |V|/2$, $|N_G(S)| \geq h|S|$ where $N_G(S) = \{j \in V \ : \ d_G(j, S) = 1\}$.

Embeddings and cuts: For our purposes, a (semi-)metric space (X, d) consists of a finite set of points $X = \{x_1, x_2, \dots, x_n\}$ and a distance function $d : X \times X \mapsto \mathbb{R}_{\geq 0}$ satisfying the following three conditions:

1. $d(x, x) = 0$, $\forall x \in X$.
2. $d(x, y) = d(y, x)$.
3. (Triangle inequality) $d(x, y) + d(y, z) \geq d(x, z)$.

An *embedding* from a metric space (X, d) to a metric space (Y, d') is a mapping $f : X \to Y$. The embedding is called a *contraction*, if

$$d'(f(x_i), f(x_j)) \leq d(x_i, x_j), \qquad \forall x_i, x_j \in X.$$

For convenience, we will only deal with contractive mappings in this paper (this is without loss of generality). A contractive mapping is said to have (worst-case) distortion Δ, if: $\sup_{i,j} \frac{d(x_i, x_j)}{d'(f(x_i), f(x_j))} \leq \Delta$. It is said to have *average* distortion β, if $\frac{\sum_{i<j} d(x_i, x_j)}{\sum_{i<j} d'(f(x_i), f(x_j))} \leq \beta$.

Note that a mapping with worst-case distortion Δ also has average distortion Δ, but not necessarily vice-versa.

Fréchet embeddings of (X, d) are a class of embeddings of $X \to \mathbb{R}^k$ into defined on the basis of distances to point sets: a co-ordinate of the embedding will be given by a map of the form $d(x_i, S) := \min_{j \in S} d(x_i, x_j)$ for some $S \subseteq X$. Note that Fréchet embeddings are always contractive in every co-ordinate.

When $X \subseteq \mathbb{R}^k$ is a ℓ_2^2 space, we will use $d(i, j) := \|x_i - x_j\|_2^2$, and $d(S, T) = \min_{i \in S, j \in T} d(i, j)$ for $S, T \subseteq [n]$. For $c \in \mathbb{R}$, $B(i, c) := \{j \ : \ d(i, j) \leq c\}$. We refer to the quantity $\frac{1}{n^2} \sum_{i,j} \|x_i - x_j\|_2^2$ as the *spread* of these points.

3 Proof of Main Theorem

3.1 Proof Outline

We prove Theorem 1.3 in two steps. First, we scale the points to lie within a ℓ_2 ball of radius 1; note that this would shrink the pairwise distances. Suppose that the points have *constant spread* after this scaling; i.e. they satisfy

$$\frac{1}{n^2} \sum_{i,j \in V} \|x_i - x_j\|_2^2 \geq \delta, \qquad \text{where } \delta = \Omega(1). \tag{3.1}$$

Since scaling does not affect the subspace rank, we continue to have $\mathrm{ssr}_\eta(X) = r$. In this case, we adapt the chaining argument from [22] to work on the *projections* $\{\Pi x_i\}_{i \in V}$ to conclude the existence of two large, Δ-separated sets for $\Delta = \Omega(1/\sqrt{r})$.

In the general case, we show that by appropriately utilizing the subspace criterion, we can either reduce it to the case of constant spread, or produce an $O(1)$ distortion Fréchet embedding by considering distances to an appropriate ℓ_2^2 ball centered at one of the points.

Let $V := [n]$. We will require the following definitions, following [3]:

▶ **Definition 3.1** (Largeness). A subset $A \subseteq V$ is β-large, if $|A| \geq \beta n$.

▶ **Definition 3.2** (Δ-separation). Subsets $L \subseteq V$ and $R \subseteq V$ are Δ-separated, if $d(L, R) \geq \Delta$.

The following lemma, implicit in [3], gives a sufficient condition for the existence of a Fréchet embedding into ℓ_1 with low average distortion.

▶ **Lemma 3.3** (Sufficient condition). *If there is a set $S \subseteq [n]$ satisfying*

$$|S| \sum_{i \notin S} d(i, S) \geq c.n^2 \tag{3.2}$$

Then, there is an embedding of the points into ℓ_1 with average distortion $1/c$.

Proof. Consider the embedding $i \mapsto d(i, S)$. Clearly, this is a Fréchet embedding, and hence a contraction. Furthermore, we have:

$$\begin{aligned}\sum_{i,j \in V} |d(i, S) - d(j, S)| &\geq \sum_{i \notin S, j \in S} |d(i, S) - 0| \\ &= |S| \sum_{i \notin S} d(i, S) \quad \geq cn^2\end{aligned}$$

Thus, the average distortion of the map is at most $1/c$. ◀

Note that the existence of two $\Omega(1)$-large, Δ-separated sets L, R would satisfy the above condition, with $S = L$ and $c = O(1/\Delta)$. The above can also be thought of as an embedding into ℓ_2, since it is one-dimensional.

3.2 The constant spread case

We will start by stating the following Proposition, which is a simple modification of Proposition 3.11 in [22]. Since the proof closely follows the original, requiring only a simple observation, we do not give it here.

▶ **Proposition 3.4** (From Proposition 3.11 in [22]). *Let $G = (V, E)$ be graph with vertex expansion $h(G) \geq 1/2$. Let $f : V \to B_2^d$ be a mapping that satisfies:*

$$\frac{1}{n^2} \sum_{i,j \in V} \|f(i) - f(j)\|_2 \geq \gamma \tag{3.3}$$

Then, there exists a pair $i, j \in V$, and constants $c_1(\gamma), c_2(\gamma)$ such that

$$\|f(i) - f(j)\|_2 \geq c_1(\gamma) \quad \text{and} \quad d_G(i, j) \leq c_2(\gamma)\sqrt{d} \tag{3.4}$$

▶ Remark. The modification only requires the observation that for any i, j with $\|f(i) - f(j)\|_2 \leq c_1(\gamma)$, and $u : \|u\|_2 = 1$, $\langle f(i) - f(j), u\rangle \leq c_1(\gamma)$. This avoids a union bound over the pairs of points in the last step of the proof, the rest of the steps being identical. Combined with the original statement of Proposition 3.11 in [22], the term $\sqrt{d}$ in the above can be replaced by $\min\left\{\sqrt{\log n}, \sqrt{d}\right\}$.

We now proceed to prove a special case of Theorem 1.3 assuming condition (3.1).

▶ **Theorem 3.5.** *Let $X = \{x_1, \ldots, x_n\}$ satisfy ℓ_2^2-triangle inequalities, with $X \subseteq B_2^d$ and $\mathrm{ssr}_\eta(X) = r$. Furthermore, suppose that*

$$\frac{1}{n^2} \sum_{ij} \|x_i - x_j\|_2^2 \geq \delta, \qquad \textit{where } \delta = \Omega(1).$$

Then there exist sets $A, B \subseteq X$, with $|A|, |B| \geq (\eta\delta/32)n$ with $d(A, B) \geq \Omega(1/\sqrt{r})$.

Proof. Let Π be the r-dimensional subspace containing an η fraction of the squared lengths of the difference vectors upon projection. Let $V = [n]$, and define $f : V \to B_2^r$ by

$$f(i) \triangleq \Pi x_i$$

Since the set X has η-subspace rank r, we have, by definition:

$$\frac{1}{n^2} \sum_{i,j \in V} \|f(i) - f(j)\|_2^2 \geq \eta\delta. \tag{3.5}$$

We will now follow the proof of Theorem 2.4 in [22], but switch to the projections where appropriate. Consider the graph $G = (V, E)$ with edges $E = \left\{\{i, j\} \ : \ \|x_i - x_j\|_2^2 \leq \frac{\kappa}{\sqrt{r}}\right\}$, where $\kappa = \kappa(\eta, \delta)$ is a constant that we will set later.

Suppose, for the sake of contradiction, that every two sets $A, B \subseteq V$ with $|A|, |B| \geq (\eta\delta/32)n$ satisfy $d(A, B) \leq \kappa/\sqrt{r}$, which implies that $d_G(A, B) \leq 1$. We use the following lemma from [22]:

▶ **Lemma 3.6** (Lemma 2.3 in [22]). *Fix $0 < \epsilon \leq \frac{1}{10}$, and let $G = (V, E)$ be a graph such that for every $X, Y \subseteq V$ satisfying $|X|, |Y| \geq \epsilon|V|$, $d_G(x, y) \leq 1$. Then there is a $U \subseteq V$ with $|U| \geq (1 - \epsilon)|V|$ with $h(G[U]) \geq \frac{1}{2}$.*

Invoking Lemma 3.6 on G yields a subset $X' \subseteq V$, with $|X'| \geq (1 - \frac{\eta\delta}{32})n$ such that $h(G[X']) \geq \frac{1}{2}$. We claim the following:

$$\frac{1}{|X'|^2} \sum_{i,j \in X'} \|f(i) - f(j)\|_2 \geq \frac{(\eta\delta)^{3/2}}{32}. \tag{3.6}$$

To see this, note that $|X' \times X'| \geq (1 - \frac{\eta\delta}{16})n^2$. Let

$$D = \left\{(i, j) \in V \times V \ : \ \|f(i) - f(j)\|_2^2 \geq \eta\delta/4\right\} .$$

Since the diameter of the unit ball is 2, in order to satisfy (3.5), we should have $|D| \geq (\eta\delta/8)n^2$. Thus, $|D \cap (X' \times X')| \geq \frac{\eta\delta}{16} n^2$. This implies that the average ℓ_2-distance in $X' \times X'$ is at least:

$$\frac{1}{n^2} |D \cap (X' \times X')| \times \sqrt{\frac{\eta\delta}{4}} \geq \frac{(\eta\delta)^{3/2}}{32}. \tag{3.7}$$

This proves (3.6).

We can now apply Proposition 3.4 to $G[X']$, and the projections $\{f(i)\}_{i \in V}$, with $\gamma = (\eta\delta)^{3/2}/32$. We infer that there exists a path in G, of $k \leq c_2(\gamma)\sqrt{r} = a(\eta, \delta)\sqrt{r}$ vertices $i_1, i_2, \ldots i_k \subseteq X'$ such that $\|f(i_1) - f(i_k)\|_2 \geq c_1(\gamma) = b(\eta, \delta)$, where $a(\eta, \delta)$ and $b(\eta, \delta)$ are constants depending on η and δ.

This implies that:

$$b^2(\eta, \delta) \overset{(a)}{\leq} \|f(i_1) - f(i_k)\|_2^2 \overset{(b)}{\leq} \|x_{i_1} - x_{i_k}\|_2^2 \overset{(c)}{\leq} \sum_{j=1}^{k-1} \left\|x_{i_j} - x_{i_{j+1}}\right\|_2^2 \overset{(d)}{\leq} a(\eta, \delta)\sqrt{r}\frac{\kappa}{\sqrt{r}}. \tag{3.8}$$

Above, (b) follows from the fact that projections can only decrease distances, (c) from the ℓ_2^2 property, and (d) from the definition of G. This is a contradiction, if we set $\kappa < \frac{b^2(\eta,\delta)}{a(\eta,\delta)}$. ◀

▶ Remark. The last chain of inequalities above is the only place where the ℓ_2^2 triangle inequalities are invoked. Without them, we could still prove a weaker statement with $O(1/r)$ separation between the large sets, since (c) would hold with an additional multiplicative factor of k by convexity.

3.3 The general case

We now extend our argument to the general case. Let us fix some notation before going to the proofs. We will take $V := [n]$, and $X = \{x_1, \ldots, x_n\}$ to satisfy the ℓ_2^2 triangle inequalities, with $\mathrm{ssr}_\eta(X) = r$. Let Π be the corresponding r-dimensional subspace. Let $f(i) := \Pi x_i$, as before. Define

$$d_f(i,j) := \|f(i) - f(j)\|_2^2 .$$

The terms $d_f(i,S)$, $d_f(S,T)$ for $S,T \subseteq V$ are defined naturally, and denote $\mathrm{diam}_f(S) \triangleq \max_{i,j\in S} d_f(i,j)$. Note that $d_f(\cdot,\cdot)$ is *not necessarily* a distance, unlike $d(\cdot,\cdot)$. However, since f is a projection map, it satisfies:

$$d(i,S) \geq d_f(i,S) \qquad \forall i \in V, \forall S \subseteq V, \tag{3.9}$$

We will also assume that X is scaled to satisfy:

$$\frac{1}{n^2} \sum_{i,j\in V} \|x_i - x_j\|_2^2 = 1 . \tag{3.10}$$

We first record a simple observation.

▶ **Observation 3.7.** *For any $i,j \in V$, and any $S \subseteq V$,*

$$d_f(i,j) \leq 3\,(d_f(i,S) + \mathrm{diam}_f(S) + d_f(j,S)).$$

Proof. Let $i^*, j^* \in S$ be such that $d_f(i,S) = d_f(i,i^*)$ and $d_f(j,S) = d_f(j,j^*)$. Since $\sqrt{d_f}$ obeys the triangle inequality, we have:

$$\begin{aligned}\left(\sqrt{d_f(i,j)}\right)^2 &\leq \left(\sqrt{d_f(i,i^*)} + \sqrt{d_f(i^*,j^*)} + \sqrt{d_f(j,j^*)}\right)^2 \\ &\leq 3(d_f(i,S) + \mathrm{diam}_f(S) + d_f(j,S))\end{aligned}$$

The last inequality follows from the convexity of the function $g(x) = x^2$, and the definition of diam_f. ◀

We now consider various cases, and show that a low average-distortion embedding exists in each case.

▶ **Lemma 3.8** (Dense Ball). *If $\exists i \in V$, with $|B(i,1/12)| \geq n/12$, then we can find an $O(1)$-average distortion embedding of X into ℓ_1.*

Proof. The proof follows the proof of a similar lemma in [3]. Let $i_0 \in V$ be such that $|B(i_0,1/12)| \geq n/12$, and let $S = B(i_0,1/12)$. Consider the embedding $i \mapsto d(i,S)$. This is a contraction. Since $\sum_{ij} \|x_i - x_j\|_2^2 = n^2$, we have:

$$\begin{aligned} n^2 &= \sum_{i,j\in V} d(i,j) \\ &\leq \sum_{i,j\in V} (d(i,S) + d(j,S)) \qquad \ldots \text{ Using } \ell_2^2 \text{ triangle inequality} \\ &= 2n\left(\sum_{i\notin S} d(i,S)\right)\end{aligned}$$

This gives us that $\sum_{i \notin S} d(i, S) \geq n/12$. Since $|S| = \Omega(n)$, Lemma 3.3 applies, and proves that the above embedding has $O(1)$ average-distortion. [2] ◀

▶ **Lemma 3.9** (Isolating a bounded ball)**.** *If there is no $i \in V$ such that $|B(i, 1/12)| \geq n/12$, then there is a $j \in V$ such that $S = B(j, 12/9)$ satisfies $|S| \geq \frac{3}{12}n$, and*

$$\sum_{i,j \in S} d(i,j) \geq \left(\frac{2}{12}\right)\left(\frac{1}{12}\right)\frac{n^2}{12}$$

Proof. Suppose we had $|B(j, 12/9)| < (3n/12)$ for every $j \in V$. Then, for any $j \in V$, we would have $|\overline{B(j, 12/9)}| > 9n/12$, which gives us that $\sum_i d(j,i) > n$. Summing over $j \in V$ contradicts (3.10).

Now, let $j_0 := \arg\max_{j \in V} |B(j, 12/9)|$, and $S := B(j_0, 12/9)$. Define the set $A = B(j_0, 12/9) \setminus B(j_0, 1/12)$. From our assumption and the preceeding argument, $|A| \geq 2n/12$. Since $|B(i, 1/12)| \leq n/12$ for every $i \in A$, we have that $\left|\overline{B(i, 1/12)} \cap A\right| \geq n/12$. This gives us:

$$\sum_{i \in A, j \in A} d(i,j) \geq \frac{2n}{12} \times \frac{1}{12} \times \frac{n}{12}.$$ ◀

In next two lemmas, assume that the precondition of Lemma 3.9 holds, i.e., there is no $i \in V$ with $|B(i, 1/12)| \geq n/12$.

▶ **Lemma 3.10.** *Let $j_0 = \arg\max_{j \in V} |B(j, 12/9)|$, and $S \triangleq B(j_0, 12/9)$. If S satisfies:*

$$\sum_{i,j \in S} d_f(i,j) \geq \frac{\eta}{600}|S|^2,$$

then there is an embedding of X into ℓ_1 with $O(\sqrt{r})$ average distortion.

Proof. Consider the map $g : V \to \mathbb{R}^d$ given by $g(i) \triangleq \sqrt{9/12} \cdot x_i$. This ensures that $g(i) \in B_2^d$ for every $i \in S$, and the mapping continues to obey the ℓ_2^2 triangle inequalities. Furthermore, from Lemma 3.9, the points in S satisfy:

$$\frac{1}{|S|^2} \sum_{i,j \in S} \|g(i) - g(j)\|_2^2 \geq \frac{9}{12} \times \frac{2}{12^3} = \Omega(1). \tag{3.11}$$

From the assumption on S, we infer that:

$$\frac{1}{|S|^2} \sum_{i,j \in S} \|\Pi g(i) - \Pi g(j)\|_2^2 \geq \frac{9}{12} \times \frac{\eta}{600}.$$

We can now invoke Theorem 3.5 on just the points in S to conclude that there exist sets $A, B \subseteq S$, such that $|A|, |B| \geq \Omega_\eta(n)$ with $d(A, B) \geq \Omega_\eta(1/\sqrt{r})$ (the scaling by a constant factor just shrinks some distances). As before, it is easy to see that A satisfies the conditions of Lemma 3.3 with $c = \Omega(1/\sqrt{r})$ and hence the mapping $h(i) \triangleq d(i, A)$ has average distortion $O(\sqrt{r})$. Note that by the ARV algorithm [3], the sets can be found with good probability by a random separating hyperplane through j_0. ◀

[2] Strictly speaking, one could do without the ℓ_2^2 triangle inequality here by adjusting the constants appropriately, as we did in Observation 3.7.

▶ **Lemma 3.11.** *Let $j_0 = \arg\max_{j\in V} |B(j, 12/9)|$, and $S \triangleq B(j_0, 12/9)$. If S satisfies:*

$$\sum_{ij\in S} d_f(i,j) \le \frac{\eta}{600}|S|^2,$$

then we can find an embedding of X into ℓ_1 with $O(1)$ average distortion.

Proof. The proof will be similar to the proof of Lemma 3.8, except for the fact that we will work with projections instead of the original vectors.

First, observe that there exists an $i_0 \in S$ such that $|B_f(i_0, \eta/24) \cap S| \ge 24|S|/25$. If not, then for every $i \in S$, we will have $\sum_{j\in S} d_f(i,j) > \frac{1}{25}|S| \times \eta/24 = \eta|S|/600$. Summing over $j \in S$ results in a contradiction to the precondition on S.

Let $T \triangleq B_f(i_0, \eta/24)$; from the preceding argument, we have $|T| = \Omega(n)$.

▶ **Claim 3.12.** $\sum_{j\notin T} d_f(j,T) \ge \eta n/12$

Proof. We know that $\sum_{i,j\in V} \|f(i) - f(j)\|_2^2 = \sum_{i,j\in V} d_f(i,j) \ge \eta n^2$. Using Observation 3.7, we can infer:

$$\begin{aligned}\eta n^2 &\le \sum_{i,j\in V} d_f(i,j)\\ &\le 3\sum_{i,j\in V} (d_f(i,T) + \mathrm{diam}_f(T) + d_f(j,T)) &&\ldots\text{Using Observation 3.7}\\ &= 3\left(2n\sum_{i\in V} d_f(i,T) + \frac{4\eta}{24}n^2\right) &&\ldots\text{ Since }\ \mathrm{diam}_f(T) \le \frac{4\eta}{24}\end{aligned}$$

This yields that $\sum_i d_f(i,T) \ge \frac{\eta}{12}n$, proving the claim. ◀

Since $|T| = \Omega(n)$, and $d(i,T) \ge d_f(i,T)$, T satisfies the conditions of Lemma 3.3. This gives us an $O(1)$ average-distortion embedding of the points into ℓ_1. ◀

We can now infer the proof of Theorem 1.3 by using the results above.

Proof of Theorem 1.3. The conditions covered in Lemmas 3.8, 3.9, 3.10 and 3.11 on the set of points $\{x_i\}_{i\in V}$ are exhaustive, and in each case yield an embedding with $O(\sqrt{r})$ average distortion. It is clear that each of these conditions can be easily checked, and the corresponding embeddings can be constructed efficiently. ◀

▶ Remark. The Hamming Cube on N points, residing in $\log N$ dimensions, and having η-subspace rank $\Omega_\eta(\log N)$ by symmetry, has two $\Omega(N)$-sized sets that are $\Omega(1/\sqrt{\log N})$ apart, and shows that the above analysis is tight up to constants.

3.4 Application to Sparsest Cut

The proof of Corollary 1.4 now follows easily, using the main result.

Proof of Corollary 1.4. Suppose $\lambda_r/n \ge \Phi_{SDP}/(1-\epsilon)$. We invoke the following result of Guruswami and Sinop [11] (stated here for the special case of *Uniform* Sparsest Cut):

▶ **Proposition 3.13** (Von-Neumann inequality [11, Theorem 3.3]). *Let $\sigma_1 \ge \sigma_2 \ge \ldots \ge \sigma_n \ge 0$ be the singular values of the matrix M with columns $\{(x_i - x_j)\}_{i<j}$. Then*

$$\frac{\sum_{t\ge r}\sigma_j^2}{\sum_{t=1}^n \sigma_j^2} \le \frac{\Phi_{SDP}}{\lambda_r(G)/n}.$$

For every $l \leq n$, we know that $\sum_{i=1}^{l} \sigma_i^2 = \sum_{i<j} \|\Pi_l(x_i - x_j)\|_2^2$, where Π_l is the subspace defined by the the top l left singular vectors of M. This immediately gives us that $\mathrm{ssr}_\epsilon(X) = r - 1$. Applying the main theorem gives us an $O(\sqrt{r})$ average distortion embedding into ℓ_1, and hence an $O_\epsilon(\sqrt{r})$ approximation to $\Phi(G)$ in this setting. ◀

▶ Remark. Under the same precondition, Guruswami and Sinop [11] give an $O(1/\epsilon)$ approximation, but by solving a SDP of size $n^{O(r)}$, using a partial solver that runs in time $2^{O(r)}\mathrm{poly}(n)$ [10]. They need to know r first, and set up the SDP and solver appropriately. The works [7, 6] give a $O(r/\epsilon^2)$ and $O(r/\epsilon)$ approximation respectively, using just the Goemans-Linial SDP; the rounding algorithms do not depend on r. Our algorithm too is independent of r, and we get a better guarantee of $O(\sqrt{r}/\epsilon)$ in this setting.

Though the precondition of the corollary may seem involved, it can easily be related back to a simpler one, as the following corollary shows (proof in Appendix A.2).

▶ **Corollary 3.14.** *If G is regular with $\lambda_r(G) \geq \epsilon$, then we can find a $O(\sqrt{r} + 1/\sqrt{\epsilon})$ approximation to the sparsest cut in G in* $\mathrm{poly}(n)$ *time.*

▶ Remark. It is clear that we get a $O(\sqrt{r})$ approximation for all graphs whose ℓ_2^2 representation always has subspace rank r. Graphs of low threshold-rank are one class of graphs that have this property.

Acknowledgements. The second named author would like to thank Amit Deshpande and Prahladh Harsha for prior useful discussions.

References

1 Noga Alon and Vitali D Milman. λ_1, isoperimetric inequalities for graphs, and superconcentrators. *Journal of Combinatorial Theory, Series B*, 38(1):73–88, 1985.

2 Sanjeev Arora, James R. Lee, and Assaf Naor. Euclidean distortion and the sparsest cut. *J. Amer. Math. Soc.*, 21:1–21, 2008. (Preliminary version in *37th STOC*, 2005). doi:10.1090/S0894-0347-07-00573-5.

3 Sanjeev Arora, Satish Rao, and Umesh V. Vazirani. Expander flows, geometric embeddings and graph partitioning. *J. ACM*, 56(2), 2009. (Preliminary version in *36th STOC*, 2004). doi:10.1145/1502793.1502794.

4 Yonatan Aumann and Yuval Rabani. An $O(\log k)$ approximate min-cut max-flow theorem and approximation algorithm. *SIAM Journal on Computing*, 27(1):291–301, 1998.

5 Jean Bourgain and Lior Tzafriri. Invertibility of large submatrices with applications to the geometry of Banach spaces and harmonic analysis. *Israel Journal of Mathematics*, 57(2):137—-224, 1987. doi:0.1007/BF02772174.

6 Amit Deshpande, Prahladh Harsha, and Rakesh Venkat. Embedding Approximately Low-Dimensional ℓ_2^2 Metrics into ℓ_1. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS), 2016*, volume 65 of *LIPIcs*, pages 10:1–10:13, 2016. doi:10.4230/LIPIcs.FSTTCS.2016.10.

7 Amit Deshpande and Rakesh Venkat. Guruswami-Sinop rounding without higher level Lasserre. In *Proc.* 17*th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX)*, volume 28 of *LIPIcs*, pages 105–114. Schloss Dagstuhl, 2014. arXiv:1406.7279, doi:10.4230/LIPIcs.APPROX-RANDOM.2014.105.

8 Shayan Oveis Gharan and Luca Trevisan. Improved ARV rounding in small-set expanders and graphs of bounded threshold rank, 2013. arXiv:1304.2060.

9 Michel X. Goemans. Semidefinite programming in combinatorial optimization. *Mathematical Programming*, 79(1):143–161, 1997.

10 Venkatesan Guruswami and Ali Kemal Sinop. Faster SDP Hierarchy Solvers for Local Rounding Algorithms. In *Proc.* 53*rd IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 197–206, 2012. doi:10.1109/FOCS.2012.58.

11 Venkatesan Guruswami and Ali Kemal Sinop. Approximating non-uniform sparsest cut via generalized spectra. In *Proc.* 24*th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 295–305, 2013. arXiv:1112.4109, doi:10.1137/1.9781611973105.22.

12 Piotr Indyk and Jirí Matoušek. Low-distortion embeddings of finite metric spaces. In Jacob E. Goodman and Joseph O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, pages 177–196. Chapman and Hall/CRC, 2nd edition, 2004. doi:10.1201/9781420035315.ch8.

13 William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference on Mondern Analysis and Probability*, volume 26 of *Contemporary Mathematics*, pages 189–206. Amer. Math. Soc., 1982. doi:10.1090/conm/026.

14 Daniel M. Kane and Raghu Meka. A PRG for Lipschitz functions of polynomials with applications to sparsest cut. In *Proc.* 45*th ACM Symp. on Theory of Computing (STOC)*, pages 1–10, 2013. arXiv:1211/1109, doi:10.1145/2488608.2488610.

15 Tsz Chiu Kwok, Lap Chi Lau, Yin Tat Lee, Shayan Oveis Gharan, and Luca Trevisan. Improved Cheeger's inequality: analysis of spectral partitioning algorithms through higher order spectral gap. In *Proc.* 45*th ACM Symp. on Theory of Computing (STOC)*, pages 11–20, 2013. arXiv:1301.5584, doi:10.1145/2488608.2488611.

16 Jean B Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

17 James R. Lee. On distance scales, embeddings, and efficient relaxations of the cut cone. In *Proc. of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, SODA, pages 92–101, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics. URL: http://dl.acm.org/citation.cfm?id=1070432.1070446.

18 Nathan Linial. Finite metric spaces: combinatorics, geometry and algorithms. In *Proc. of the ICM, Beijing*, volume 3, pages 573–586, 2002. arXiv:math/0304466.

19 Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

20 Avner Magen and Mohammad Moharrami. On the nonexistence of dimension reduction for ℓ_2^2 metrics. In *Proc.* 20*th Annual Canadian Conf. on Comp. Geom.*, 2008. URL: http://cccg.ca/proceedings/2008/paper37full.pdf.

21 Jirí Matoušek. Embedding finite metric spaces into normed spaces. In *Lectures on Discrete Geometry*, Graduate Texts in Mathematics, chapter 5, pages 355–400. Springer, 2002. doi:10.1007/978-1-4613-0039-7_15.

22 Assaf Naor, Yuval Rabani, and Alistair Sinclair. Quasisymmetric embeddings, the observable diameter, and expansion properties of graphs. *Journal of Functional Analysis*, 227(2):273–303, 2005.

23 Assaf Naor and Robert Young. The integrality gap of the Goemans–Linial SDP relaxation for Sparsest Cut is at least a constant multiple of $\Omega(\sqrt{\log n})$. In *Proc.* 49*th ACM Symp. on Theory of Computing (STOC)*, 2017. arXiv:1704.01200.

24 Luca Trevisan. Lecture notes on expansion, sparsest cut, and spectral graph theory, 2011. Available online. URL: http://www.eecs.berkeley.edu/~luca/books/expanders.pdf.

25 Joel A. Tropp. Column subset selection, matrix factorization, and eigenvalue optimization. In *Proc.* 20*th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 978–986, 2009. URL: http://dl.acm.org/citation.cfm?id=1496770.1496876, arXiv:0806.4404.

A Appendix

A.1 Ruling out a worst-case distortion bound of $O(\sqrt{\mathrm{ssr}_\eta(X)})$

We give a simple example of why one cannot hope to prove a worst-case distortion bound like Goemans' result, using the notion of subspace rank. Suppose that a certain point set X satisfies the ℓ_2^2 inequalities, and has *worst-case* distortion $\Omega(D)$ for embedding into ℓ_1. It is known that there exists such an X with $D = \Omega(\sqrt{\log n})$ [23]. Without loss of generality, let X be scaled to satisfy $\sum_{i,j} \|x_i - x_j\|_2^2 = n^2$, and $\|x_1 - x_2\|_2^2 = \max_{i,j} \|x_i - x_j\|_2^2$. Consider the set Y which has X, along with $C - 1$ additional copies of x_1 and x_2[3]. Clearly, Y satisfies the ℓ_2^2 triangle inequalities. Further, Y has η-subspace rank of 1 for a large enough C: the sum of all squared distances is at most $C + (C^2 - C)\|x_1 - x_2\|_2^2$, and the sum of squared distances along the direction $x_1 - x_2$ is at least $C^2 \|x_1 - x_2\|_2^2$. However, embedding Y with *worst*-case distortion $O(1)$ into ℓ_1 would contradict the lower bound on embedding X into ℓ_1.

A.2 Proof of Corollary 3.14

Proof (Of Corollary 3.14). The proof follows by using a combination of two algorithms, depending on how λ_r compares to $\Phi_{SDP}(G)$. Suppose that G is 1-regular by scaling the edge weights, without loss of generality, and let $X = \{x_1, \ldots, x_n\}$ be the optimal SDP solution. If $\Phi_{SDP} \geq \epsilon/100n$, then there is one co-ordinate of the SDP solution with objective value at least $\epsilon/100n$. In this case, running the Cheeger rounding algorithm [1, Lemma 2.1] (see also [24, Section 2.4] for an exposition) on this co-ordinate would output a cut of sparsity $O(\sqrt{\epsilon}/n) \leq O\left(\Phi_{SDP}(G)/\sqrt{\epsilon}\right)$.

If $\Phi_{SDP} \leq \epsilon/100n$ then we have $\lambda_r/n \geq 100\Phi_{SDP}$. Applying Corollary 1.4 with $\epsilon = 99/100$ gives us an $O(\sqrt{r})$ average-distortion embedding into ℓ_1, and hence an $O(\sqrt{r})$ approximation to $\Phi(G)$ in this setting. Thus, the best of the two cuts will be a $O(\sqrt{r} + 1/\sqrt{\epsilon})$ approximation to $\Phi(G)$. ◀

[3] Technically, we are dealing with semi-metrics, and hence distinct points may overlap.

When Are Welfare Guarantees Robust?*†

Tim Roughgarden[1], Inbal Talgam-Cohen[2], and Jan Vondrák[3]

1 **Stanford University, Stanford, CA, USA**
`tim@cs.stanford.edu`

2 **Hebrew University of Jerusalem, Jerusalem, Israel**
`inbal.talgamcohen@mail.huji.ac.il`

3 **Stanford University, Stanford, CA, USA**
`jvondrak@stanford.edu`

Abstract

Computational and economic results suggest that social welfare maximization and combinatorial auction design are much easier when bidders' valuations satisfy the "gross substitutes" condition. The goal of this paper is to evaluate rigorously the folklore belief that the main take-aways from these results remain valid in settings where the gross substitutes condition holds only approximately. We show that for valuations that pointwise approximate a gross substitutes valuation (in fact even a linear valuation), optimal social welfare cannot be approximated to within a subpolynomial factor and demand oracles cannot be simulated using a subexponential number of value queries. We then provide several positive results by imposing additional structure on the valuations (beyond gross substitutes), using a more stringent notion of approximation, and/or using more powerful oracle access to the valuations. For example, we prove that the performance of the greedy algorithm degrades gracefully for near-linear valuations with approximately decreasing marginal values; that with demand queries, approximate welfare guarantees for XOS valuations degrade gracefully for valuations that are pointwise close to XOS; and that the performance of the Kelso-Crawford auction degrades gracefully for valuations that are close to various subclasses of gross substitutes valuations.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Valuation (set) functions, gross substitutes, linearity, approximation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.22

1 Introduction

Welfare maximization in combinatorial auctions is a central problem in both theory and practice, and is perhaps the most well-studied problem in algorithmic game theory (e.g., [7]). The problem is: given a set M of distinct items and descriptions of, or oracle access to, the valuation functions $v_1, \ldots, v_n$ of n bidders (each a set function from bundles to values), determine the partition $S_1, \ldots, S_n$ of items that maximizes the social welfare $\sum_{i=1}^{n} v_i(S_i)$.

This paper focuses on the purely algorithmic and approximation aspects of welfare maximization. Many possibility and impossibility results for efficiently computing or approximating the maximum social welfare are known, as a function of the set of allowable bidder valuations. Very roughly, the current state of affairs can be summarized by a trichotomy:

* A full version of the paper is available at `https://arxiv.org/abs/1608.02402`.

† This work is supported by NSF grants CCF-1215965 and CCF-1524062, and has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 708935.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 22; pp. 22:1–22:23
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

(i) when the valuations satisfy "gross substitutes," then exact welfare maximization is easy;
(ii) when the valuations are "complement-free" but not necessarily gross substitutes (e.g., subadditive), exact welfare maximization is hard but constant-factor approximations are possible; and
(iii) with sufficiently general valuations, even approximate welfare maximization is hard.[1]

The rule of thumb that "substitutes are easy, complements are hard" has its origins in the economic literature on multi-item auction design. For example, simple and natural ascending auctions converge to a welfare-maximizing Walrasian equilibrium (i.e., to a market-clearing price vector and allocation) whenever all bidders' valuations satisfy gross substitutes [24], but when this condition is violated a Walrasian equilibrium does not generally exist [18, 28]. Similarly, the VCG mechanism has a number of desirable properties (like revenue monotonicity) when bidders' valuations satisfy gross substitutes, but not in general otherwise [2].

Thus both computational and economic results suggest that social welfare maximization and combinatorial auction design are much easier when bidders' valuations satisfy the gross substitutes condition. More generally, in settings with both substitutes and complements, the folklore belief in the field is that simple auction formats should produce allocations with near-optimal social welfare if and only if the substitutes component is in some sense "dominant."[2] For over twenty years, there has been a healthy (and high-stakes) debate over whether or not the ideal case of gross substitutes valuations should guide combinatorial auction design in realistic settings.[3]

The goal of this paper is to evaluate rigorously the folklore belief that the main take-aways from the study of gross substitutes valuations are robust, i.e., remain valid in settings where the substitutes condition holds only approximately. We are interested both in possibility/impossibility results for achieving robustness (Section 3), and also in understanding the robustness of standard algorithms and auctions for welfare maximization (Sections 4–5, Appendix D). Our goal relates to a wider research agenda on the robustness or stability of "nice" classes of valuations: For different notions of closeness to nice valuations (in our case gross substitutes or even simply linear valuations), do fundamental optimization tasks (in our case welfare maximization) maintain their algorithmic guarantees? We consider a standard notion of pointwise closeness (Section 3), and subsequently add to it "semantic" closeness (Section 4).[4] This research agenda is motivated by inherent mathematical interest in classes of set functions and their stability [10, 14], as well as by applications like data-driven optimization in which parameters of the optimization problem such as valuations are approximately estimated from data [5, 4, 43, 19, 3].

1.1 Our Results

We first consider arguably the most natural notion of "approximate gross substitute valuations," namely valuations that are pointwise within a $1 + \epsilon$ factor of some gross substitutes

[1] "Gross substitutes" means that a bidder's demand for an item can only increase as prices of other items increase; see Section 2.1 for a formal definition.

[2] For example, Bykowsky et al. [8] write: "In general, synergies across license valuations complicate the auction design process. Theory suggests that a 'simple' (i.e., non-combinatorial) auction will have difficulty in assigning licenses efficiently in such an environment."

[3] Ausubel et al. [1] write: "A contentious issue in the design of the Federal Communications Commission (FCC) auctions of personal communications services (PCS) licenses concerned the importance of synergies. If large synergies are prevalent among the licenses being offered, then the simultaneous ascending auction mechanism the FCC adopted, which does not permit all-or-nothing bids on sets of licenses, might be expected to perform poorly."

[4] What does it *mean* for a valuation to belong to a nice class? If a valuation approximately maintains the meaningful properties of the class, we say it is *semantically* close to it.

valuation. Do the laudable properties of gross substitutes valuations degrade gracefully as ϵ increases? At this level of generality, the answer is negative: even for valuations that are pointwise close to *linear* (affine) valuations, we prove that the social welfare cannot be approximated to within a subpolynomial factor using a subexponential number of value queries, and that demand oracles cannot be approximated in any useful sense by a subexponential number of value queries. When the gross substitutes condition holds exactly, a demand query can be implemented using a polynomial number of value queries (see [36]), and the welfare maximization problem can be solved exactly with a polynomial number of such queries (see [33]). We conclude that there is no sweeping generalization of the properties of gross substitutes valuations to approximations of such valuations, and that any positive result must impose additional structure on the valuations (beyond gross substitutes), use a more stringent notion of approximation, and/or use more powerful oracle access to the valuations.

We next consider positive results in the value oracle model. Our main result here is a proof that the (optimal) performance of the greedy algorithm degrades gracefully for valuations that are close to linear functions, provided these valuations are also approximately *submodular* (in a stronger than pointwise sense). (As noted above, some assumption beyond near-linearity is necessary for any positive result.) The standard arguments for proving approximation bounds for the greedy algorithm (e.g. [26]) do not imply this result, and we develop a new analysis for this purpose.

We then consider welfare maximization with demand oracles,[5] and find that welfare guarantees tend to be more robust in this model. First, the *value* of optimal social welfare can be approximated using demand queries within a factor of $\gamma_{\mathcal{C}} + \epsilon$, whenever the valuations are pointwise ϵ-close to a class $\mathcal{C}$ such that the integrality gap of the "configuration LP" is $\gamma_{\mathcal{C}}$. Since the configuration LP is the primary vehicle for developing approximation algorithms in the demand oracle model, we recover pointwise robustness essentially for all known approximation results in the demand oracle model (in terms of the optimal value). Another question, though, is whether an allocation achieving good welfare can be found in a computationally-efficient way. For some classes of valuations, we show that this is possible (and thus we achieve a $(1-\epsilon)$-approximation for valuations ϵ-close to linear, and a $(1-1/e-\epsilon)$-approximation for valuations ϵ-close to XOS). We remark that no extra assumption of near-submodularity is required here; this highlights another difference between the value and demand oracle models.

Another approach to finding an optimal allocation assuming the gross substitutes property is the Kelso-Crawford algorithm, also known as the *tâtonnement* procedure. In general, this procedure requires demand queries and thus also falls within the umbrella of the demand oracle model. Here we show that the performance of the Kelso-Crawford algorithm degrades smoothly for certain classes of functions, namely for valuations close to linear, close to unit-demand with $\{0,1\}$ values, and more generally close to transversal valuations (rank functions of a partition matroid). Unfortunately the Kelso-Crawford algorithm is not going to be a universally robust solution for general gross substitutes, either. As we show, its performance degrades discontinuously for valuations close to unit-demand (with unrestricted values), a seemingly minor extension of the cases above where we showed positive results.

In addition we show a counterexample to an approach based on Murota's cycle canceling algorithm, the remaining known way to solve the welfare maximization problem under the gross substitutes property.

[5] As noted above, with valuations that only satisfy gross substitutes approximately, demand oracles are substantially more powerful than value oracles.

In summary, the idea that approximation guarantees for various classes of valuations should degrade gracefully under small deviations from the class should be viewed with skepticism. Our negative results do not imply that that the folklore belief that "close to substitutes is easy" is wrong, but they do imply that there is no "generic reason" for why this might be the case. Our positive results show that robust guarantees can still be obtained in certain cases, but typically this requires additional care and often new ideas on top of known techniques.

1.2 Related Work and Organization

There is growing interest in the algorithmic game theory literature in the class of gross substitutes valuations prominent in the economic literature [36, 21, 37]. One reason for this is that welfare maximization for gross substitutes has deep mathematical and algorithmic roots – the simple subcase of unit-demand valuations already subsumes bipartite matching [39]. The main algorithmic techniques include ascending-price auctions [24], combinatorial cycle-canceling algorithms that utilize discrete convexity [31, 32], and linear programming [6, 33]. Submodular valuations are a strict superclass of gross substitutes, and for these the following is known: exact welfare maximization is hard, a 2-approximation can be achieved greedily [26], and continuous greedy – the optimal approximation algorithm in the value oracle model – achieves a $(1 - 1/e)$-approximation [46, 25]. In the demand query model, a 2-approximation is achieved via an ascending-price auction [17], and the $1 - 1/e$ barrier can be broken [15].

Not much previous work studies the extent to which welfare maximization is robust to small deviations. The closest result to ours is a theorem of [19], ruling out a constant-factor approximation when maximizing valuations that are pointwise ϵ-close to submodular, rather than to gross substitutes. Our Proposition 4 strengthens this negative result to apply to gross substitutes valuations and in fact even to valuations that are close to linear. Related lower bounds appear e.g. in [30, 38, 20], often relying on the same approach of hiding structure in a seemingly "nice" set function. The challenge in our case is to allow the function to appear linear.

Several related works differ from ours in the model of deviation, valuation classes considered, and/or setting: [19] studies welfare maximization with submodular valuations and random rather than adversarial noise. [29] introduces a distance metric from matroid rank functions in a single-parameter rather than combinatorial setting. [13] and references within study deviations from submodularity captured by graphical representations. [27] tests the convergence and performance of a heuristic approach based on cycle-canceling for subclasses of submodular valuations.

Finally, it is interesting to compare our work to [23], which studies divisible items and budgets rather than indivisible items and quasi-linear utilities. [23] defines an *approximate weak gross substitutes* property and shows that with this property a known auction mechanism guarantees each bidder approximately his demand (in a different sense than our Definition 22(4)). The conclusion of [23], by which "markets do not suddenly become intractable if they slightly violate the weak gross substitutes property", is quite different from ours, showing an intriguing discrepancy between the divisible and indivisible models.

Organization. After preliminaries in Section 2, Section 3 presents two cautionary tales as to what can go wrong when simple valuations undergo smalls perturbations. We then establish positive results in the value oracle model (Section 4) and in the demand oracle model (Section 5), and conclude in Section 6. We defer some proofs to Appendices A–C. Negative results for standard algorithms appear in Appendix D and in the full version [40].

2 Preliminaries

A *combinatorial auction* (or *market*) includes a set of n players (or *bidders*) N and a set of m indivisible items M. We call a subset $S \subseteq M$ of items a *bundle*. For an ordered set of items $S = (s_1, s_2, \ldots, s_k)$ and for $j \in [k]$, we use the notation S_j to denote the "prefix" subset $\{s_1, \ldots s_{j-1}\}$ of items that appear before s_j (in particular, $S_1 = \emptyset$).

Every player i has a *valuation* $v_i : 2^M \to \mathbb{R}_+$ over bundles. Valuations are assumed to be monotone unless stated otherwise, i.e., $v(S) \leq v(T)$ for every two bundles $S \subseteq T$. Valuations are not necessarily normalized, i.e., $v(\emptyset)$ is not always equal to zero (normalization is not without loss of generality in our model of perturbation). [6] For every two bundles S, T we denote by $v(S \mid T)$ the *marginal* value $V(S \cup T) - v(T)$ of S given T.

An *allocation* $\mathcal{S}$ is a partition of the items in M into n bundles $S_1, \ldots, S_n$ where S_i is the allocation of player i. In a *full* allocation every item is allocated. The social efficiency of $\mathcal{S}$ is measured by its *welfare* $W(\mathcal{S}) = \sum_{i=1}^n v_i(S_i)$.

A *price vector* $p \in \mathbb{R}_+^m$ is a vector of item prices. We denote by $p(S)$ the aggregate price $\sum_{j \in S} p(j)$ of the bundle S. Given p, each player wishes to maximize his *quasi-linear utility*, i.e., to receive a bundle S maximizing $v_i(S) - p(S)$, which we say is *in demand*. If the player is already allocated a bundle T, he wishes to add a bundle $S \subseteq (M \setminus T)$ maximizing $v_i(S \mid T) - p(S)$, which we say is in demand *given* T. A bundle S is *individually rational (IR)* for a player i if $v_i(S) \geq p(S)$; it is *strongly IR* if every subset $T \subseteq S$ is IR for i.

A full allocation $\mathcal{S}$ and price vector p form a *Walrasian equilibrium* if for every player i, S_i is in i's demand given p. By the *1st welfare theorem*, $\mathcal{S}$ maximizes welfare.

Since valuations are in general of exponential size in m, the standard assumption is that they are accessed via *value oracles*, which return the value of any bundle upon query. In Sections 3.2 and 5 we discuss *demand oracles*, which given a price vector p, return a bundle in the player's demand given p. We allow p to include negative prices so that a demand oracle can return a bundle in the player's demand *given a previous allocation* T.

2.1 Valuation Classes

A valuation ℓ is *linear* (also known as affine) if there exists a vector $(l_1, \ldots, l_m) \geq 0$ and a scalar $c \geq 0$ such that $\ell(S) = c + \sum_{j \in S} l_j$ for every bundle S; it is *additive* if $c = 0$. A valuation r is *unit-demand* if there exists a vector $(\rho_1, \ldots, \rho_m) \geq 0$ such that $r(S) = \max_{j \in S} \rho_j$ for every bundle S.

Let $\mathcal{M} = (M, \mathcal{I})$ be a matroid over the ground set of items M, where $\mathcal{I}$ is the family of *feasible* bundles. [7] A maximal feasible set is called a *basis*, and the matroid *rank function* maps bundles to the cardinality of their largest feasible subset. Given a vector $(w_1, \ldots, w_m) \geq 0$ of item weights, the corresponding *weighted* rank function maps bundles to the weight of their heaviest feasible subset. A valuation r is an unweighted (resp., weighted)

[6] A natural example of a non-normalized valuation is that of a firm over sets of workers, where some positions are already filled [34].

[7] A set system $\mathcal{M} = (M, \mathcal{I})$ is a matroid if the following properties hold: (i) $\mathcal{I}$ is non-empty; (ii) $\mathcal{I}$ is downward-closed, that is, if $S \subseteq T$ and $T \in \mathcal{I}$ then $S \in \mathcal{I}$; (iii) for every $S, T \in \mathcal{I}$ such that $|S| < |T|$, there exists $t \in T \setminus S$ such that $S \cup \{t\} \in \mathcal{I}$ (see, e.g., [35]).

matroid rank function if there exists a matroid $\mathcal{M}$ such that r is its (weighted) rank function. Both additive and unit-demand valuations are types of weighted matroid rank functions.

A valuation v is *gross substitutes* if for every two price vectors $p \leq q$, for every bundle S in demand given p, there exists a bundle T in demand given q, which contains every item $j \in S$ for which $q(j) = p(j)$. [8] A valuation v is *submodular* if for every two bundles $S \subseteq T$ and item $j \notin T$, $v(j \mid S) \geq v(j \mid T)$, i.e., the marginal value of j is decreasing. A valuation v is *XOS* (also known as *fractionally subadditive*) if there exist additive valuations $a_1, a_2, \ldots$ such that $v(S) = \max_k a_k(S)$ for every bundle S. It is well-known that gross substitutes $\subset$ submodular $\subset$ XOS.

2.2 Welfare Maximization

There are three main algorithmic approaches to finding a welfare-maximizing allocation in combinatorial auctions with gross substitutes: (1) auction-based, (2) LP-based, and (3) combinatorial. For linear valuations there is also (4) greedy. The first two approaches use demand queries and run in polynomial time, while the latter two use value queries and run in strongly-polynomial time. We describe here Approaches (1) and (4); see also Appendix A.

In the Kelso-Crawford auction (Approach (1), Algorithm 1 in Appendix A), in each round an arbitrary player adds to his existing allocation a bundle in his demand given his current bundle and the current prices (with a δ-increase in the prices of items not currently in his allocation). The algorithm terminates when no player wants to add to his allocation. By the definition of gross substitutes, at any point in the algorithm every player's allocation is a subset of a bundle in his demand, and so:

▶ **Proposition 1** ([24]). *For gross substitutes valuations, Algorithm 1 converges to a (welfare-maximizing) Walrasian equilibrium as $\delta \to 0$.*

The greedy algorithm (Approach (4), Algorithm 2 in Appendix A) finds a value-maximizing bundle subject to a feasibility constraint $\mathcal{I}$ on the bundles;[9] this is a problem to which welfare maximization is well-known to reduce (see proof of Corollary 16). The greedy algorithm starts with an empty bundle, and in each iteration adds the item with maximum marginal value among all items that maintain feasibility, breaking ties arbitrarily.

▶ **Proposition 2** ([16]). *If v is linear and $\mathcal{I}$ is such that $\mathcal{M} = (M, \mathcal{I})$ is a matroid, then Algorithm 2 is optimal.*

3 Two Cautionary Tales

Do the nice properties of a valuation class degrade gracefully as one moves outside the class? This section describes two cautionary tales demonstrating that the answer can subtly depend on the notion of "being close," on the tractable class under consideration, and on the model of access to the valuations. Arguably the most natural general notion of "closeness" is pointwise:[10]

[8] The intuition behind this definition is revealed by the Kelso-Crawford algorithm below.

[9] The constraint $\mathcal{I}$ is simply a family of feasible bundles; we assume it is given by a feasibility oracle.

[10] We use relative error in the interest of scale-invariance (the "units" in which valuations are specified do not matter). However it is interesting to note that our negative results in this section would hold even for additive (rather than linear) valuations in a model of additive (rather than multiplicative) error, as studied e.g. by [10].

▶ **Definition 3.** A valuation $\tilde{v}$ is ϵ-*close* to submodular (or linear, gross substitutes, etc.), if there is a submodular (or linear, gross substitutes, etc.) valuation v such that $v(S) \le \tilde{v}(S) \le (1+\epsilon)v(S)$ for all bundles S.

3.1 Close-to-additive vs. Close-to-linear Valuations

We next show that approximate welfare maximization is hard for valuations that are ϵ-close to linear (a restrictive subclass of gross substitutes valuations).

▶ **Proposition 4.** *Given value oracles for valuations ϵ-close to linear, no algorithm using a subexponential number of queries can approximate the value of optimal social welfare within a factor better than polynomial in m, n.*

Proof Sketch. Let $|M| = m = an$ and let $(A_1, A_2, \dots, A_n)$ be a random partition of M such that $|A_i| = a$. We define linear valuations as follows: $\ell_i(S) = \epsilon + \frac{1}{a}|S \cap A_i|$. We also define $w_i(S) = (1+\epsilon)\epsilon + \frac{1}{an}|S|$. Note that if we do not know the partition $(A_1, \dots, A_n)$, which is randomized, $\ell_i(S)$ will be close to its expectation which is $w_i(S)$. Let us define the following valuations v_i:

- If $\big||S \cap A_i| - \frac{1}{n}|S|\big| \le \epsilon^2 a$, then $v_i(S) = w_i(S)$.
- If $\big||S \cap A_i| - \frac{1}{n}|S|\big| > \epsilon^2 a$, then $v_i(S) = \ell_i(S)$.

Note that in the first case, we have $v_i(S) \ge (1+\epsilon)\epsilon + \frac{1}{a}(|S \cap A_i| - \epsilon^2 a) = \ell_i(S)$ and $v_i(S) \le (1+\epsilon)\epsilon + \frac{1}{a}(|S \cap A_i| + \epsilon^2 a) \le (1+2\epsilon)\epsilon + \frac{1}{a}|S \cap A_i| \le (1+2\epsilon)\ell_i(S)$. So v_i is 2ϵ-close to ℓ_i.

By Chernoff bounds, for a fixed query S, the probability that $\big||S \cap A_i| - \frac{1}{n}|S|\big| > \epsilon^2 a$ is $e^{-\Omega(\epsilon^4 a)}$. Hence, with high probability we are always in the first case above, the returned value depends only on $|S|$ and hence the algorithm does not learn any information about A_i. Therefore (by standard arguments), it would require exponentially many queries to find any set such that $\big||S \cap A_i| - \frac{1}{n}|S|\big| > \epsilon a$ and hence distinguish whether the input valuations are v_i or w_i.

The optimal solution under v_i is $S_i = A_i$ which gives welfare $\sum_{i=1}^n v_i(A_i) > \frac{1}{a}\sum_{i=1}^n |A_i| = n$, while the optimal welfare under w_i is $(1+\epsilon)\epsilon n + 1$. So the approximation factor cannot be better than $(1+\epsilon)\epsilon + \frac{1}{n}$.

Note that we can set $\epsilon = 1/a^{1/4-\delta} = (n/m)^{1/4-\delta}$ and the high probability statements still hold. Therefore, we can push the hardness factor to $\max\{(n/m)^{1/4-\delta}, 1/n\}$. ◀

This hardness result relies strongly on the value oracle model – things are quite different in the demand oracle model, as we discuss in Section 3.2 below.

Proposition 4 is a startling contrast to the case of valuations that are ϵ-close to *additive* valuations, which differ only by requiring the empty set to have value 0. Here, approximate welfare maximization is easy (a proof appears for completeness in Appendix B).

▶ **Proposition 5.** *Welfare maximization can be solved within a factor of $1+\epsilon$ for ϵ-close to additive valuations.*

The proof of Proposition 5 makes the more general point that, whenever the approximating "nice" valuation $\tilde{v}$ can be (approximately) recovered from the given valuation v, then welfare approximation guarantees carry over (applying an off-the-shelf approximation algorithm to the valuations $\tilde{v}$). As Proposition 4 makes clear, however, in many cases it is not possible to efficiently reconstruct an approximating "nice" valuation, even under the promise that such a valuation exists.

3.2 Value Queries vs. Demand Queries

One remarkable property of gross substitutes valuations is that there is no difference between the value oracle and demand oracle models: Demand queries can be simulated efficiently by value queries (via the greedy algorithm), and hence any algorithm in the demand oracle model can also be implemented in the value oracle model ([36] and references within). Unfortunately, this is no longer true for valuations that are ϵ-close to gross substitutes, or even ϵ-close to linear.

▶ **Proposition 6.** *Given a value oracle to a valuation ϵ-close to linear, answering demand queries requires an exponential number of value queries.*

This can be proved by a construction similar to the one above. However, let us instead present an indirect argument which shows more.

▶ **Lemma 7.** *For any class of valuations $\mathcal{C}$ such that the integrality gap of the configuration LP is $\gamma \geq 1$, it is possible to estimate the optimal social welfare within a multiplicative factor of $(1+\epsilon)\gamma$ for valuations that are ϵ-close to $\mathcal{C}$ in the demand oracle model.*

Proof. Consider the configuration LP:

$$\begin{array}{rc} \max & \sum_{i=1}^{n} \sum_{S \subseteq [m]} v_i(S) x_{i,S} : \\ \forall j \in [m]; & \sum_{i=1}^{n} \sum_{S : j \in S} x_{i,S} \leq 1, \\ \forall i \in [n]; & \sum_{S \subseteq [m]} x_{i,S} = 1, \\ & x_{i,S} \geq 0. \end{array}$$

Let us denote by $\mathrm{LP}(\mathbf{v})$ and $\mathrm{OPT}(\mathbf{v})$ the LP optimum and optimal social welfare, respectively, under valuations $\mathbf{v}$. The assumption is that for valuations $\mathbf{v} \in \mathcal{C}$, $\mathrm{OPT}(\mathbf{v}) \leq \mathrm{LP}(\mathbf{v}) \leq \gamma \cdot \mathrm{OPT}(\mathbf{v})$. Let us solve the LP for valuations $\tilde{\mathbf{v}}$ that are pointwise ϵ-close to valuations $\mathbf{v} \in \mathcal{C}$. (This is possible since we assume that we have demand oracles for $\tilde{\mathbf{v}}$, which gives a separation oracle for the dual; see [33].) We have $v_i(S) \leq \tilde{v}_i(S) \leq (1+\epsilon) v_i(S)$ for each i and S. Therefore, the same inequalities hold for the LP optimum as well as optimal social welfare, and we obtain $\mathrm{OPT}(\tilde{\mathbf{v}}) \leq \mathrm{LP}(\tilde{\mathbf{v}}) \leq (1+\epsilon)\,\mathrm{LP}(\mathbf{v}) \leq (1+\epsilon)\gamma \cdot \mathrm{OPT}(\mathbf{v}) \leq (1+\epsilon)\gamma \cdot \mathrm{OPT}(\tilde{\mathbf{v}})$. ◀

Since the configuration LP is known to be integral for gross substitutes valuations (see e.g. [45]), we obtain the following.

▶ **Corollary 8.** *For valuations ϵ-close to gross substitutes, it is possible to estimate the optimal social welfare to within a multiplicative factor of $1+\epsilon$ in the demand oracle model.*

This implies Proposition 6: a simulation of demand queries would allow us to approximate social welfare within $1+\epsilon$ for valuations ϵ-close to gross substitutes and as a special case ϵ-close to linear. We know from Proposition 4 that this (and even much weaker approximations) would require exponentially many value queries. Actually we obtain a stronger statement: It is not possible to answer demand queries via value queries for ϵ-close to linear valuations even approximately, under *any notion of approximation* that would be useful for approximating the optimal social welfare. This is because, again, such a simulation would lead to a $(1+\epsilon)$-approximation of social welfare via value queries, which Proposition 4 rules out.

4 Positive Results for Restricted Valuation Classes

Motivated by the hardness results in the previous section for valuations ϵ-close to linear in the value oracle model, this section considers stronger notions of closeness. In Section 4.1 we define a semantic notion of *marginal* closeness. In Section 4.2 we analyze the greedy algorithm for ϵ-close to linear valuations, but this time to avoid the impossibility results we assume they are also marginally close to submodular. Intuitively, such valuations enable positive results by approximately maintaining some of the semantic properties of linear valuations (namely submodularity).

4.1 Marginal Closeness

Recall that a pointwise approximation of a valuation is a set function with approximately the same output as the valuation for every input. A natural strengthening of this is to have the set function's discrete derivatives approximate those of the valuation – i.e., the items' marginal values. For every item j, its marginal value $v(j \mid \cdot)$ is a set function mapping S to $v(j \mid S)$, and together these set functions encode important properties of the valuation. For example, linear valuations can be characterized as valuations for which the marginal of every item is a constant function; submodular valuations can be characterized as valuations for which the marginal of every item is a non-increasing function; and unweighted matroid rank functions can be characterized as valuations for which the marginal of every item is a non-increasing zero-one function [41].

We remark that even without any strengthening, pointwise ϵ-closeness has the following useful implication regarding the "closeness" of the marginal values of bundles:

▶ **Observation 9.** *For every valuation v that is ϵ-close to a valuation v', and for every two bundles $S, T \subseteq M$, $v'(S \mid T) - \epsilon v'(T) \leq v(S \mid T) \leq v'(S \mid T) + \epsilon v'(S \cup T)$.*

Proof. By the definition of ϵ-closeness, $v(S \mid T) = v(S \cup T) - v(T) \geq v'(S \cup T) - (1+\epsilon)v'(T) = v'(S \mid T) - \epsilon v'(T)$. The upper bound follows similarly. ◀

Observation 9 will be useful in proving some of our positive results (see Sections 4.2 and 5.2). Yet its guarantee is relatively weak: the "error terms" depend on $v'(T)$ or $v'(S \cup T)$, and so can be large if these terms are large. We now define the stronger notion of marginal pointwise approximation:[11]

▶ **Definition 10.** A valuation v is *marginal-ϵ-close* to a class $\mathcal{C}$ of marginals (constant, non-increasing, etc.), if for every item j the corresponding marginal value function $v(j \mid \cdot)$ is (pointwise) ϵ-close to a function $g_j \in \mathcal{C}$.

Let us now see what semantic properties follow from this definition. Consider the class of valuations that are marginal-ϵ-close to decreasing. It turns out that this coincides with a previously-studied class of approximately submodular valuations (see Appendix B for missing proofs).

▶ **Definition 11** ([26]). Let $\alpha \geq 1$. A valuation v is *α-submodular* if for every two bundles $S \subseteq T$ and item $j \notin T$, $\alpha v(j \mid S) \geq v(j \mid T)$.

▶ **Proposition 12.** *A valuation v is marginal-ϵ-close to the class of decreasing functions if and only if it is $(1+\epsilon)$-submodular.*

[11] This notion is equivalent to having erroneous oracle access to marginal values, an idea mentioned in [9].

Since we are interested in closeness notions that maintain semantic properties, in the next section we consider what happens when the ϵ-close to linear valuations from Section 3 are also α-submodular. Such valuations are well-studied in the literature as they contain submodular valuations with bounded curvature, which were introduced by [11] and arise frequently in the optimization and learning literatures ([42, 44] and references within):

▶ **Definition 13** ([11, 44])**.** A valuation v has *curvature* $c \in [0, 1]$ if for every item j and bundle S such that $j \notin S$, $v(j \mid S) \geq (1 - c)v(j \mid \emptyset)$.

▶ **Proposition 14.** *An α-submodular valuation v with curvature c is $\frac{\alpha-1+c}{1-c}$-close to a linear valuation.*

4.2 The Greedy Algorithm

Recall that the greedy algorithm (Algorithm 2) is optimal for linear valuations; in this section and in Appendix B we prove the following robustness theorem, whose implication for welfare maximization is stated in Corollary 16.

▶ **Theorem 15.** *Let v be an α-submodular valuation that is ϵ-close to a linear valuation ℓ. Let $\mathcal{M} = (M, \mathcal{I})$ be a matroid of rank k (represented by an independence oracle). The greedy algorithm returns an independent set $S \in \mathcal{I}$ such that $v(S) \geq \frac{1-3\epsilon}{\alpha} v(S^*)$, where $S^* \in \arg\max_{I \subseteq \mathcal{I}} v(I)$.*

▶ **Corollary 16.** *In a market with α-submodular valuations $v_1, \ldots, v_n$ that are ϵ-close to linear valuations $\ell_1, \ldots, \ell_n$, there is a polynomial time algorithm with value access that finds an allocation with $\frac{1-3\epsilon}{\alpha}$-approximately optimal welfare.*

Proof. As in [26], we define a valuation v over player-item pairs, such that for a set S of such pairs, $v(S) = \sum_i v_i(S_i)$ where S_i is the set of items paired with player i in S. We show that v is also α-submodular and ϵ-close to linear: It is ϵ-close to linear since $\sum_i \ell_i(S_i) \leq v(S) \leq (1 + \epsilon) \sum_i \ell_i(S_i)$, and the sum of linear valuations is linear. It is α-submodular since for every $S \subseteq T$ and $(i, j) \notin T$, $\alpha v((i, j) \mid S) = \alpha v_i(j \mid S_i) \geq v_i(j \mid T_i) = v((i, j), T)$. The proof follows by applying Theorem 15 to v and to the partition matroid $\mathcal{M}$ over player-item pairs, which allows each item to be paired with no more than one player. ◀

The following lemma (whose proof appears in Appendix B) relates the sum of marginals to the linear contribution, and is applied in the proof of Theorem 15. Recall that v is ϵ-close to the linear valuation ℓ, and that for an ordered set of items $X = (x_1, x_2, \ldots, x_k)$, X_j denotes the prefix $\{x_1, \ldots x_{j-1}\}$.

▶ **Lemma 17.** *Let X and Z be two disjoint sets. X is ordered and has m_X items. Let $Y_1, \ldots, Y_{m_X}$ be sets such that $Y_j \subseteq X_j \cup Z$ for every j. Then $\alpha \sum_{j=1}^{m_X} v(x_j \mid Y_j) \geq \sum_{j=1}^{m_X} \ell(x_j) - \epsilon \ell(Z)$.*

Before proving Theorem 15, we highlight the difference between our proof and the standard item-by-item analysis of the greedy algorithm (see, e.g., [26]). The standard analysis shows that greedily choosing the next item maintains the optimal value up to a small error. However this does not provide a good approximation factor in our case because of the distortion caused by the pointwise errors. For example, greedy may choose to include an item j from the optimal solution, but at the particular stage in which j is included its marginal value may not be high relative to its linear contribution (due to the error term in the marginal value shown in Observation 9). This suggests analyzing multiple items at a time so that the error terms will average out. We split the items chosen by greedy into two sets – those that coincide with the optimal solution and the rest – and analyze each set as a whole in order to establish the claimed approximation ratio.

Proof of Theorem 15. Denote by S the independent set that greedy returns given the valuation v and matroid $\mathcal{M}$, ordered by the order in which the items were greedily added. Denote by S^* an independent set with optimal value $v(S^*)$ subject to the matroid constraint. Due to monotonicity we can assume without loss of generality that the sizes of both S and S^* are k, i.e., that both sets are bases.

Let $P = \{j \mid s_j \in S \cap S^*\}$ be the positions in S of the items that also appear in S^*, and let Q be the remaining positions in S. Order S^* such that the items in $S \cap S^*$ are in positions P. The rest of the items in S^* are ordered among the remaining positions in S^* in the following way: By the exchange property of matroids, there exists a bijection f from $S \setminus S^*$ to $S^* \setminus S$, such that for every position $j \in Q$, $S \setminus \{s_j\} \cup \{f(s_j)\}$ is independent [41]. For every $j \in Q$ we set $s^*_j = f(s_j)$. This ensures that s^*_j can be added to S_j without violating feasibility. We use the notation $S(P)$ and $S(Q)$ for the set of items in S in positions P and Q, respectively; similarly for $S^*(P)$ and $S^*(Q)$.

By the definition of greedy, and because adding s^*_j to S_j maintains feasibility for every $j \in Q$, it holds that $v(s_j \mid S_j) \geq v(s^*_j \mid S_j)$ for every $j \in [k]$. Thus

$$v(S) \geq v(\emptyset) + \sum_{j \in P} v(s_j \mid S_j) + \sum_{j \in Q} v(s^*_j \mid S_j). \tag{1}$$

We now invoke Lemma 17 twice. By Lemma 17 instantiated with $X = S^*(Q)$ and $Z = S$ (note these are indeed disjoint), and using that $S_j \subseteq S$, we get that

$$\alpha \sum_{j \in Q} v(s^*_j \mid S_j) \geq \sum_{j \in Q} \ell(s^*_j) - \epsilon \ell(S). \tag{2}$$

By Lemma 17 instantiated with $X = S(P)$ and $Z = S(Q)$ (these are also disjoint), and using that $S_j \subseteq S(Q) \cup \{s_{p_1}, \ldots, s_{p_{j-1}}\}$, we get that

$$\alpha \sum_{j \in P} v(s_j \mid S_j) \geq \sum_{j \in P} \ell(s_j) - \epsilon \ell(S) = \sum_{j \in P} \ell(s^*_j) - \epsilon \ell(S). \tag{3}$$

Combining (1), (2) and (3), we get that $\alpha v(S) \geq \alpha v(\emptyset) + \sum_{j=1}^{k} \ell(s^*_j) - 2\epsilon \ell(S)$. Since v is ϵ-close to ℓ and S^* is optimal for v, $\ell(S) \leq v(S) \leq v(S^*)$. Again using v's closeness to ℓ, $\alpha v(\emptyset) \geq \alpha c \geq c$. Putting these together we get that $\alpha v(S) \geq \ell(S^*) - 2\epsilon v(S^*) \geq v(S^*)/(1+\epsilon) - 2\epsilon v(S^*) \geq (1 - 3\epsilon) v(S^*)$, as required. ◀

5 The Demand Oracle Model

In this section we achieve positive results by considering a stronger oracle model.

5.1 Rounding the Configuration LP

By Lemma 7 and Corollary 8, we already know that polynomial-time *estimation* of the optimal welfare is robust in the demand oracle model. A different question is whether *finding* an allocation given a fractional solution of the configuration LP is also robust. Here we observe that in some cases, existing rounding techniques yield the result that we want.

The following proposition does not assume submodularity:

▶ **Proposition 18.** *For combinatorial auctions in the demand oracle model, there is a polynomial time algorithm that finds:*

- *A $(1-\epsilon)$-approximately optimal allocation, if the valuations are ϵ-close to linear;*
- *A $(1-1/e-\epsilon)$-approximately optimal allocation, if the valuations are ϵ-close to XOS (or in particular ϵ-close to submodular).*

Proof. We solve the configuration LP using demand queries. Let $\tilde{v}_i$ denote the true valuations and v_i the linear/XOS valuations that the $\tilde{v}_i$ are close to.

In the case of valuations close to linear, we round the fractional solution by allocating each item j to player i with probability $y_{ij} = \sum_{S:j\in S} x_{i,S}$. Call the resulting random sets R_i. For the underlying linear valuations v_i, it is clearly the case that $\mathbb{E}[v_i(R_i)] = \sum_S x_{i,S} v_i(S)$. Therefore, $\sum_i \mathbb{E}[\tilde{v}_i(R_i)] \geq \sum_i \mathbb{E}[v_i(R_i)] = \sum_{i,S} x_{i,S} v_i(S) \geq \frac{1}{1+\epsilon}\,\mathrm{LP}(\tilde{v}_i) \geq (1-\epsilon)\,\mathrm{OPT}$.

For valuations close to XOS, we allocate a set S tentatively to player i with probability $x_{i,S}$, and then we use the contention resolution technique to resolve conflicts [12, 15]. This technique has the property that conditioned on requesting an item j, a player receives it with conditional probability at least $1-1/e$. Denote by R_i the random set that player i receives after contention resolution. Using the fractional subadditivity of XOS functions, we obtain that $\mathbb{E}[v_i(R_i)] \geq (1-1/e)\sum_S x_{i,S} v_i(S)$. Hence, similarly to the case above, $\sum_i \mathbb{E}[\tilde{v}_i(R_i)] \geq \sum_i \mathbb{E}[v_i(R_i)] \geq (1-1/e)\sum_{i,S} x_{i,S} v_i(S) \geq \frac{1-1/e}{1+\epsilon}\,\mathrm{LP}(\tilde{v}_i) \geq (1-1/e-\epsilon)\,\mathrm{OPT}$. ◀

5.2 Positive Results for Kelso-Crawford

The Kelso-Crawford algorithm (Algorithm 1) uses demand queries in order to find, in each iteration, the items to add to a player's existing bundle. Here we show that it can work well for markets in which the valuations are approximately submodular (maintain semantic closeness) and ϵ-close to simple subclasses of gross substitutes. Our high-level approach is to show that Kelso-Crawford finds an allocation which achieves approximately optimal welfare, as well as a price vector that is "approximately stabilizing" (in a sense made precise in Definition 22, which may be of independent interest).

▶ Remark. For simplicity, our analysis of the Kelso-Crawford algorithm shall treat prices as if raised continuously rather than discretely. This means that the approximation factors we state in this section hold up to a small additive error.[12]

We first establish that the Kelso-Crawford algorithm works well for the class of α-submodular and ϵ-close to linear valuations, as studied above in Section 4.

▶ **Theorem 19** (Linear). *In a market with α-submodular valuations that are ϵ-close to linear valuations, the Kelso-Crawford algorithm finds an allocation with $\frac{1}{\alpha+2\epsilon}$-approximately optimal welfare (up to a vanishing error).*

On the negative side and somewhat surprisingly, we find this positive result does not extend even to ϵ-close to unit-demand (and still α-submodular) valuations (see Appendix D and the full version):

▶ **Proposition 20.** *For every $\epsilon \leq 1$, there exists a market with $O(1/\epsilon)$ players whose submodular valuations are ϵ-close to unit-demand, for which the Kelso-Crawford algorithm with adversarial ordering of the players finds an allocation with $\approx 2/3$ of the optimal welfare.*

[12] If the prices are increased by discrete δ increments, then the additive error is of order $O(mn\delta)$.

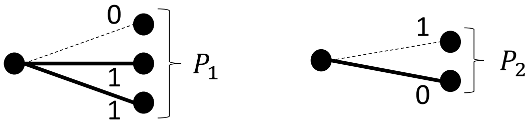


Figure 1 A graphical representation of an unweighted transversal valuation r. The edges correspond to items and have $\{0,1\}$ weights; they are partitioned in this case into two parts P_1, P_2. The value that r attributes to the subset of solid bold edges in this example is 1, since this is the weight of the maximum-weight matching within the subset.

We thus proceed to consider the *unweighted* version of unit-demand valuations, and more generally the direct sums of such valuations – called *unweighted transversal* valuations. Such valuations arise in natural economic environments, e.g. in the context of labor markets. [13] For these valuations we establish in Theorem 21 that Kelso-Crawford maintains good welfare guarantees.

▶ **Theorem 21** (Transversal). *In a market with α-submodular valuations that are ϵ-close to unweighted transversal valuations, the Kelso-Crawford algorithm finds an allocation with $\frac{1}{\alpha(1+3\epsilon)^2}$-approximately optimal welfare (up to a vanishing error).*

In more detail, a valuation r is an unweighted unit-demand valuation if $v(S) = \max_{j\in S} r(j)$ for every bundle S and $r(j) \in \{0,1\}$ for every item j. A valuation r is an unweighted transversal valuation if there exists a partition $\mathcal{P} = (P_1, \dots, P_k)$ of the items such that $v(S) = \sum_{P\in\mathcal{P}} \max_{j\in S\cap P} r(j)$ for every bundle S, and $r(j) \in \{0,1\}$ for every item j. An equivalent definition in terms of matchings in graphs is the following: r is an unweighted transversal valuation if it can be represented by a bipartite graph whose vertices on one side of the graph all have degree 1, and whose edges M have $\{0,1\}$ edge weights. The value for a bundle of edges S is the weight of the maximum matching in the bipartite graph induced by S – see Figure 1 for an example.

5.2.1 Biased Walrasian Equilibrium: Definition and Properties

The proofs of Theorems 19 and 21 follow from the welfare guarantees of a solution concept that we call a "biased" Walrasian equilibrium (Definition 22 and Proposition 23), combined with an appropriate lemma showing that Kelso-Crawford converges to such an equilibrium for the relevant class of valuations (Lemma 27 for Theorem 19, and Lemma 29 for Theorem 21). Recall that the standard proof establishing the optimality of Kelso-Crawford for gross substitutes relies on showing convergence to a Walrasian equilibrium. Unfortunately in our case it does not hold – even approximately – that Kelso-Crawford allocates to each player a bundle in his demand. Instead, our proofs will define and utilize a different notion of an approximate Walrasian equilibrium.

▶ **Definition 22.** Consider a market with n players, and let $\mu \in [0,1]$. A full allocation $\mathcal{S}$ and a price vector p form a *μ-biased Walrasian equilibrium* if there exists $\mu' \in [\mu, 1]$ such that for every alternative allocation T and for every player i,

$$\frac{\mu'}{\mu} v_i(S_i) - p(S_i) \ge \mu' v_i(T_i) - p(T_i). \tag{4}$$

[13] Consider for example a firm wishing to hire a team of several workers, each with a different specialization, from a pool of specialist workers who are each either acceptable to the firm or not. The firm's valuation over workers is unweighted transversal.

▶ **Proposition 23** (Approximate first welfare theorem.). *The welfare of a μ-biased Walrasian equilibrium is a μ-approximation to the optimal welfare.*

Proof. Let $\mathcal{S}^*$ be an optimal allocation. Without loss of generality we can assume that S^* is a full allocation. By summing up Inequality (4) over all players, we get $(\mu'/\mu)W(\mathcal{S}) - p(\mathcal{S}) \geq \mu' W(\mathcal{S}^*) - p(\mathcal{S}^*)$. Since both $\mathcal{S}, \mathcal{S}^*$ are full allocations, $p(\mathcal{S}) = p(\mathcal{S}^*)$. We conclude that $W(\mathcal{S}) \geq \mu \, \mathrm{OPT}$. ◀

A possible economic interpretation of a μ-biased Walrasian equilibrium is that its allocation and prices induce market stability under the *endowment effect* (see, e.g., [22]). This effect is modeled as an increase of factor μ'/μ in a player's value for the bundle he owns, and a decrease of factor μ' in his value for bundles he does not own.

5.2.2 Convergence to Biased Walrasian Equilibrium

To give intuition for establishing convergence to biased Walrasian equilibrium (see Lemmas 27 and 29), let us compare the case of linear valuations to that of ϵ-close to linear (and α-submodular) valuations. The analysis of Kelso-Crawford for the linear case is simple – every player ends up with the items for which he has the highest marginal value and can afford to pay the highest price. In the close-to-linear case, an item could end up belonging to the wrong player for two reasons: at some point, given his current allocation (which is subject to changes), either (1) the right player has a low marginal value for the item, or (2) the wrong player has a high marginal value for it. The latter issue is resolved by submodularity, but the former could drive Kelso-Crawford to performance that is bounded away from optimal, as demonstrated in Proposition 20. Thus the crux of the convergence proofs is to show this cannot happen for the classes of valuations in question. For the close-to-linear case (Lemma 27), we achieve this by considering the marginal value of all "under-valued" items together if we were to add them to the player's final allocation. The transversal case (Lemma 29) is more involved since we cannot analyze only the addition of items to the player's final allocation – we need to also consider swapping out items from the final allocation and replacing them with others. This case is deferred to Appendix C.

We conclude this subsection with a generalization of a well-known invariant property of Kelso-Crawford which holds for submodular valuations [17]. The following generalization to α-submodular valuations is used below to prove Lemmas 27 and 29.

▶ **Definition 24.** Let v be a valuation and p be a price vector. A bundle S is *α-IR* (α-individually rational) for v given p if $\alpha v(S) \geq p(S)$. A bundle S is *strongly α-IR* for v given p if $\alpha v(T) \geq p(T)$ for every $T \subseteq S$.

▶ **Lemma 25.** *Consider a market with a player whose valuation v is α-submodular. Then the Kelso-Crawford algorithm maintains the following invariant: the player's allocation throughout the algorithm is strongly α-IR.*

Proof. Since removing items from the player's allocation maintains the strong α-IR property, the only case we need to check is when a bundle T is added to the player's current allocation S. We know that T maximizes the player's utility given S and the current price vector p. We show this by induction. In the base case all prices are 0 and so the invariant holds. Now assume for contradiction that $S \cup T$ is not strongly α-IR, i.e., there exists a set $S' \cup T'$ where $S' \subseteq S$ and $T' \subseteq T$ such that $\alpha v(S' \cup T') < p(S') + p(T')$. By the induction assumption, $\alpha v(S') \geq p(S')$, and so it must be the case that $\alpha v(T' \mid S') < p(T')$. So using α-submodularity we can write the utility $v(T \mid S) - p(T)$ as $v(T \setminus T' \mid S) + v(T' \mid S \cup T \setminus T') - p(T \setminus T') - p(T') \leq v(T \setminus T' \mid$

$S) + \alpha v(T' \mid S') - p(T \setminus T') - p(T') < v(T \setminus T' \mid S) - p(T \setminus T')$. Thus adding $T \setminus T'$ to S adds more to the utility than adding T to S, contradiction. ◀

The following corollary of Lemma 25 generalizes a result of [17, Propositions 1-2].

▶ **Corollary 26.** *In a market with α-submodular valuations, the Kelso-Crawford algorithm finds an allocation with $(1+\alpha)$-approximately optimal welfare (up to a vanishing error).*

5.3 Kelso-Crawford and Close-to-Linear Valuations

▶ **Lemma 27.** *In a market with α-submodular valuations $v_1, \dots, v_n$ that are ϵ-close to linear valuations $r_1, \dots, r_n$, the Kelso-Crawford algorithm converges to a $\frac{1}{\alpha+2\epsilon}$-biased Walrasian equilibrium.*

Proof. We can assume without loss of generality that Kelso-Crawford returns a full allocation (see proof of Theorem 21). From now until the end of the proof, fix a player i. Let S_i be player i's allocation and let p be the price vector at termination of the KC algorithm. Let T_i be an alternative bundle for player i. We show that

$$v_i(S_i) - p(S_i) \geq v_i(T_i) - p(T_i) - (2\epsilon + \alpha - 1)\, v_i(S_i). \tag{5}$$

This is sufficient to complete the proof, since by summing up over all players and rearranging, we get

$$(2\epsilon + \alpha) \sum_i v_i(S_i) - \sum_i p(S_i) \geq \sum_i v_i(T_i) - \sum_i p(T_i),$$

and so $\mu \geq 1/(\alpha + 2\epsilon)$.

It remains to prove Inequality (5). For simplicity we omit i from the notation. Without loss of generality, assume that in the last round of Kelso-Crawford, the player added to his existing bundle, which we denote by B, a bundle which maximizes his utility given B and given the price vector p. So the player's utility at termination $v_i(S_i) - p(S_i)$ is at least $u_p(B \cup T)$. We use the notation $B' = B \setminus T$, $T' = T \setminus B$, and $C = B \cap T$. We can now write the lower bound on the player's utility as

$$u_p(B \cup T) = v(\emptyset) + v(B' \mid \emptyset) + v(C \mid B') + v(T' \mid B) - p(B') - p(C) - p(T'). \tag{6}$$

By Lemma 25, B is α-strongly IR. So $v(B' \mid \emptyset) + (\alpha - 1)v(B' \mid \emptyset) \geq p(B')$. By Observation 9, $v(C \mid B') \geq \sum_{j \in C} \ell(j) - \epsilon v(B)$. Again by Observation 9, $v(T' \mid B) \geq \sum_{j \in T'} \ell(j) - \epsilon v(B)$. Plugging in these inequalities to (6) and using v's ϵ-closeness to ℓ we get

$$\begin{aligned} u_p(B \cup T) &\geq c - (\alpha - 1)v(B' \mid \emptyset) + \sum_{j \in T} \ell(j) - p(T) - 2\epsilon v(B) \\ &\geq \ell(T) - p(T) - (2\epsilon + \alpha - 1)\, v(B). \end{aligned}$$

Inequality (5) follows, and this completes the proof. ◀

Proof of Theorem 19. The proof follows directly from Proposition 23 combined with Lemma 27. ◀

6 Conclusion

Let us summarize the findings of this paper: The robustness of results for various classes of valuations should not be assumed without further investigation. The default assumption should be that positive results are *not* robust and may break down abruptly when small deviations from the class in question are introduced. Robust results can be attained, but they are surprisingly challenging to obtain even in simple cases. Most importantly, it matters how “closeness” to a class is defined, and what the other attributes of the variant of the problem are (oracle model, class of valuations).

Acknowledgements. A preliminarily version appeared as [40].

References

1 Lawrence M. Ausubel, Peter Cramton, R. Preston McAfee, and John McMillan. Synergies in wireless telephony: Evidence from the broadband PCS auctions. *Journal of Economics and Management Strategy*, 6(3):497–527, 1997.

2 Lawrence M. Ausubel and Paul R. Milgrom. The lovely but lonely Vickrey auction. In Peter Cramton, Yoav Shoham, and Richard Steinberg, editors, *Combinatorial Auctions*, chapter 1, pages 57–95. MIT Press, Boston, MA, USA, 2006.

3 Eric Balkanski, Aviad Rubinstein, and Yaron Singer. The limitations of optimization from samples. Working paper, 2016.

4 Alexandre Belloni, Tengyuan Liang, Hariharan Narayanan, and Alexander Rakhlin. Escaping the local minima via simulated annealing: Optimization of approximately convex functions. In *Proceedings of the 28th Annual Conference on Learning Theory*, pages 240–265, 2015.

5 Dimitris Bertsimas and Aurélie Thiele. *Robust and Data-Driven Optimization: Modern Decision Making Under Uncertainty*, chapter 5, pages 95–122. INFORMS PubsOnline, 2014. TutORials in Operations Research.

6 Sushil Bikhchandani and John W. Mamer. Competitive equilibrium in an exchange economy with indivisibilities. *Journal of Economic Theory*, 74(2):385–413, 1997.

7 Liad Blumrosen and Noam Nisan. Combinatorial auctions. In Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani, editors, *Algorithmic Game Theory*, chapter 11. Cambridge University Press, 2007.

8 M. M. Bykowsky, R. J. Cull, and J. O. Ledyard. Mutually destructive bidding: The FCC auction design problem. *Journal of Regulatory Economics*, 17(3):205–228, 2000.

9 Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

10 Flavio Chierichetti, Abhimanyu Das, Anirban Dasgupta, and Ravi Kumar. Approximate modularity. In *Proceedings of the 56th Symposium on Foundations of Computer Science*, pages 1143–1162, 2015.

11 Michele Conforti and Gérard Cornuéjols. Submodular set functions, matroids and the greedy algorithm: Tight worst-case bounds and some generalizations of the Rado-Edmonds theorem. *Discrete Applied Mathematics*, 7(3):251–274, 1984.

12 Uriel Feige. On maximizing welfare when utility functions are subadditive. *SIAM J. Comput.*, 39(1):122–142, 2009.

13 Uriel Feige, Michal Feldman, Nicole Immorlica, Rani Izsak, Brendan Lucier, and Vasilis Syrgkanis. A unifying hierarchy of valuations with complements and substitutes. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 872–878, 2014.

14 Uriel Feige, Michal Feldman, and Inbal Talgam-Cohen. Approximate modularity revisited. In *Proc. of the 49th Annual ACM Symp. on Theory of Computing*, 2017. To appear.

15 Uriel Feige and Jan Vondrák. The submodular welfare problem with demand queries. *Theory of Computing*, 6(1):247–290, 2010.

16 M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions – II. *Mathematical Programming Study*, 8:73–87, 1978.

17 Hu Fu, Robert Kleinberg, and Ron Lavi. Conditional equilibrium outcomes via ascending price processes with applications to combinatorial auctions with item bidding. In *Proceedings of the 13th ACM Conference on Economics and Computation*, page 586, 2012. Extended abstract.

18 Faruk Gul and Ennio Stacchetti. Walrasian equilibrium with gross substitutes. *Journal of Economic Theory*, 87:95–124, 1999.

19 Avinatan Hassidim and Yaron Singer. Submodular optimization under noise. Manuscript, 2016.

20 John William Hatfield, Nicole Immorlica, and Scott Duke Kominers. Testing substitutability. *Games and Economic Behavior*, 75(2):639–645, 2012.

21 Justin Hsu, Jamie Morgenstern, Ryan Rogers, Aaron Roth, and Rakesh Vohra. Do prices coordinate markets? To appear in STOC 2016, 2016.

22 Daniel Kahneman, Jack L. Knetsch, and Richard H. Thaler. Experimental tests of the endowment effect and the Coase theorem. *Journal of Political Economy*, 98(6):1325–1348, 1990.

23 Chinmay Karande and Nikhil R. Devanur. Computing market equilibrium: Beyond weak gross substitutes. In *Proceedings of the 3rd International Workshop on Internet and Network Economics*, pages 368–373, 2007.

24 A. Kelso and V. Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica*, 50(6):1483–1504, 1982.

25 Subhash Khot, Richard J. Lipton, Evangelos Markakis, and Aranyak Mehta. Inapproximability results for combinatorial auctions with submodular utility functions. *Algorithmica*, 52(1):3–18, 2008.

26 Benny Lehmann, Daniel Lehmann, and Noam Nisan. Combinatorial auctions with decreasing marginal utilities. *Games and Economic Behavior*, 55:270–296, 2006.

27 Takanori Maehara and Kazuo Murota. Valuated matroid-based algorithm for submodular welfare problem. *Annals of Operations Research*, 229:565–590, 2015.

28 Paul R. Milgrom. Putting auction theory to work: The simultaneous ascending auction. *Journal of Political Economy*, 108(2):245–272, 2000.

29 Paul R. Milgrom. The substitution metric and the performance of clock auctions, 2015. Talk at Simons Insitute, available at `https://simons.berkeley.edu/talks/paul-milgrom-10-13`.

30 Vahab S. Mirrokni, Michael Schapira, , and Jan Vondrák. Tight information-theoretic lower bounds for welfare maximization in combinatorial auctions. In *Proceedings of the 9th ACM Conference on Economics and Computation*, pages 70–77, 2008.

31 Kazuo Murota. Valuated matroid intersection I: Optimality criteria. *SIAM J. Discrete Math.*, 9(4):545–561, 1996.

32 Kazuo Murota. Valuated matroid intersection II: Algorithms. *SIAM J. Discrete Math.*, 9(4):562–576, 1996.

33 Noam Nisan and Ilya Segal. The communication requirements of efficient allocations and supporting prices. *Journal of Economic Theory*, 129:192–224, 2006.

34 Michael Ostrovsky and Renato Paes Leme. Gross substitutes and endowed assignment valuations. *Theoretical Economics*, 2014.

35 J. G. Oxley. *Matroid Theory*. Oxford, 1992.

36 Renato Paes Leme. Gross substitutability: An algorithmic survey. Working paper, 2014.

37 Renato Paes Leme and Sam Chiu-wai Wong. Computing Walrasian equilibria: Fast algorithms and economic insights. In *Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 632–651, 2017.

38 Christos H. Papadimitriou, Michael Schapira, and Yaron Singer. On the hardness of being truthful. In *Proceedings of the 49th Symposium on Foundations of Computer Science*, pages 250–259, 2008.

39 Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, 2000.

40 Tim Roughgarden, Inbal Talgam-Cohen, and Jan Vondrák. When are welfare guarantees robust? In *2nd Algorithmic Game Theory and Data Science Workshop*, July 2016. Full version available at `https://arxiv.org/abs/1608.02402`.

41 A. Schrijver. *Combinatorial Optimziation: Polyhedra and Efficiency*. Springer, 2003.

42 Dravyansh Sharma, Amit Deshpande, and Ashish Kapoor. On greedy maximization of entropy. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1330–1338, 2015.

43 Yaron Singer and Jan Vondrák. Information-theoretic lower bounds for convex optimization with erroneous oracles. In *Proceedings of the 28th Neural Information Processing Systems Conference*, 2015.

44 Maxim Sviridenko, Jan Vondrák, and Justin Ward. Optimal approximation for submodular and supermodular optimization with bounded curvature. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1134–1148, 2015.

45 Rakesh V. Vohra. *Mechanism Design: A Linear Programming Approach*. Econometric Society Monographs, 2011.

46 Jan Vondrák. Optimal approximation for the submodular welfare problem in the value oracle model. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 67–74, 2008.

A Standard Algorithms for Welfare-Maximization

Algorithms 1 and 2 describe the ascending price and the greedy approaches to welfare maximization.

ALGORITHM 1: The Kelso-Crawford ascending-price auction (formulated as an algorithm).

```
Input: Player valuations $v_1, \dots, v_n$ represented by demand oracles; a parameter $\delta > 0$
Output: An allocation $\mathcal{S}$ and a price vector $p$
$p := 0$ and $\mathcal{S} := \emptyset$;    % Initialization
while there exists a player $i$ and a non-empty bundle $D_i$ such that $D_i$ is in demand given prices
  $p + \delta\vec{\mathbb{1}}_{j \notin S_i}$ and current allocation $S_i$, do
    $S_i := S_i \cup D_i$;
    $S_{i'} := S_{i'} \setminus D_i$ for every $i' \neq i$;
    $p(j) := p(j) + \delta$ for every $j \in D_i$;
end
```

ALGORITHM 2: Greedy maximization of value subject to a feasibility constraint.

```
Input: A valuation $v$ represented by a value oracle; a feasibility constraint represented by a
       feasiblity oracle
Output: A bundle $S$
$S := \emptyset$;   % Initialization
while there exists an item $j \notin S$ such that $S \cup \{j\}$ is feasible do
    Let $j^*$ be an item that maximizes $v(j^* \mid S)$ among all items $j \notin S$ such that $S \cup \{j\}$ is
      feasible;
    $S := S \cup \{j^*\}$;
end
```

B Missing Proofs from Sections 3 and 4

Proof of Proposition 5. An additive valuation has the form $v(S) = \sum_{j \in S} a_j$. By assumption, $a_j \leq v(\{j\}) \leq (1+\epsilon)a_j$ for every singleton j. Hence we can determine the coefficients a_j within a factor of $1+\epsilon$ and run welfare maximization on the resulting additive functions. Each item simply goes to the highest bidder, and we lose a factor of at most $1+\epsilon$ due to the errors in determining a_j. ◀

Proof of Proposition 12. For every $S \subseteq T$ and $j \notin T$, let g_j be the decreasing function to which $v(j|\cdot)$ is ϵ-close. Then $(1+\epsilon)v(j|S) \geq (1+\epsilon)g_j(S) \geq (1+\epsilon)g_j(T) \geq v(j|T)$, showing that v is $(1+\epsilon)$-submodular. The converse follows from Observation 28. ◀

▶ **Observation 28.** *If a valuation v is α-submodular then it is marginal-ϵ-close for $\epsilon = \alpha - 1$ to the class of decreasing functions.*

Proof Sketch. Fix an item j. We want to show that the set function $v(j \mid \cdot)$ is ϵ-close to a decreasing function g_j. We define g_j as follows. For every bundle S, there is a range $[\frac{1}{1+\epsilon}v(j \mid S), v(j \mid S)]$ to which $g_j(S)$ must belong. Furthermore, for g_j to be decreasing, for every S and every subset $T \subseteq S$, the upper-bound on the range of $g_j(T)$ is also an upper-bound on the range of $g_j(S)$. We claim that: (1) If there is a non-zero range for $g_j(S)$ for every bundle S, then by picking the highest value in the range to be $g_j(S)$, we have defined a decreasing g_j such that $v(j \mid \cdot)$ is ϵ-close to it; (2) If there is a bundle S with negative range then we have found a contradiction to α-submodularity of v. Indeed, if this is the case then we have found a subset T such that $v(j \mid T) < v(j \mid S)/1+\epsilon = v(j \mid S)/\alpha$. This concludes the proof. ◀

Proof of Proposition 14. Define a linear valuation ℓ as follows: $\ell(S) = (1-c)(v(\emptyset) + \sum_{j=1}^{|S|} v(s_j \mid \emptyset))$ for every bundle S. Using that v has curvature c, $v(S) = v(\emptyset) + \sum_{j=1}^{|S|} v(s_j|S_j) \geq (1-c)(v(\emptyset) + \sum_{j=1}^{|S|} v(s_j|\emptyset)) = \ell(S)$. Using that v is α-submodular, $v(S) = v(\emptyset) + \sum_{j=1}^{|S|} v(s_j|S_j) \leq v(\emptyset) + \alpha \sum_{j=1}^{|S|} v(s_j|\emptyset) \leq \frac{\alpha}{1-c}\ell(S)$. Thus v is ϵ-close to ℓ for ϵ such that $1+\epsilon = \frac{\alpha}{1-c}$. ◀

Proof of Lemma 17.

$$\begin{aligned} \alpha \sum_{j=1}^{m_X} v(x_j \mid Y_j) &\geq \sum_{j=1}^{m_X} v(x_j \mid Z \cup X_j) && (7)\\ &= v(X \mid Z) && (8)\\ &\leq \ell(X \mid Z) - \epsilon \ell(Z), && (9) \end{aligned}$$

where (7) follows since $Y_j \subseteq Z \cup X_j$ and since v is α-submodular, (8) follows by the disjointness of X and Z, and (9) follows since v is ϵ-close to ℓ and by Observation 9. ◀

C Kelso-Crawford and Close-to-Transversal Valuations

▶ **Lemma 29.** *In a market with α-submodular valuations $v_1, \dots, v_n$ that are ϵ-close to unweighted transversal valuations $r_1, \dots, r_n$, the Kelso-Crawford algorithm converges to a $\frac{1}{\alpha(1+3\epsilon)^2}$-biased Walrasian equilibrium.*

Proof. By monotonicity we can assume without loss of generality that all items are allocated at price 0 in the first step of the Kelso-Crawford algorithm, and once an item is allocated then it remains allocated throughout. Thus without loss of generality, Kelso-Crawford returns a full allocation as required by the definition of a μ-biased Walrasian equilibrium.

From now until the end of the proof, fix a player i. For simplicity we omit i from the notation. Let $\mathcal{P} = (P_1, \dots, P_k)$ be the partition of the items corresponding to the unweighted transversal valuation r, to which the player's valuation v is ϵ-close. For an item j, let $P(j)$ denote the part to which this item belongs. We say that a part P is *represented* in a bundle X if X contains at least one item $j \in P$ with value $r(j) = 1$.

Towards proving the guarantee in Inequality (4), let S be the bundle allocated to the player by the Kelso-Crawford algorithm, and let T be an alternative bundle. Let $P(S)$ (resp., $P(T)$) be the parts represented in S (resp., T). Let $S' = \{j \in S \mid P(j) \in P(S) \cap P(T)\}$ be the set of items in S that belong to parts represented both in S and in T, and similarly $T' = \{j \in T \mid P(j) \in P(S) \cap P(T)\}$.

▶ **Claim 30.** $r(S') = r(T')$.

Proof of Claim 30. The value assigned to a set by valuation r is the number of parts represented in it. By definition, the same parts are represented in S' and in T'. ◀

We now relate the prices of S' and T' according to the price vector p with which the Kelso-Crawford algorithm terminated. In particular we show that the price of S' cannot be too high in comparison to the price of T'.

▶ **Claim 31.** $p(S') \leq p(T') + 3\alpha\epsilon r(S')$.

The proof of Claim 31 appears below. We use the above claims to complete the proof of Lemma 29. Let $S'' = S \setminus S'$ and $T'' = T \setminus T'$. We know that when the Kelso-Crawford algorithm terminates, the player cannot improve his utility by adding T'' to his allocation S, and so:

$$\begin{aligned} v(S) - p(S) &\geq v(S \cup T'') - p(S \cup T'') \\ &= v(S'') + v(S' \mid S'') + v(T'' \mid S) - p(S'') - p(S') - p(T''). \end{aligned} \tag{10}$$

By α-submodularity Lemma 25 applies and S'' is α-IR, therefore

$$v(S'') - p(S'') \geq (1 - \alpha)v(S''). \tag{11}$$

The marginal value assigned by r to a bundle X given a bundle Y is the number of parts represented in X but not in Y. Therefore $r(S' \mid S'') = r(S')$ and $r(T'' \mid S) = r(T'')$. By Observation 9 and since $r(S') = r(T')$ (Claim 30),

$$v(S' \mid S'') \geq \quad r(S' \mid S'') - \epsilon r(S'') \quad \geq r(S') - \epsilon r(S) = r(T') - \epsilon r(S); \tag{12}$$

$$v(T'' \mid S) \geq \quad r(T'' \mid S) - \epsilon r(S) \quad = r(T'') - \epsilon r(S). \tag{13}$$

Plugging (11)–(13) into (10), and using that $r(T')+r(T'')=r(T)$ and $p(S')\le p(T')+3\alpha\epsilon r(S')$ (Claim 31),

$$\begin{aligned} v(S)-p(S) &\ge (1-\alpha)v(S'')+r(T)-2\epsilon r(S)-p(T)-3\alpha\epsilon r(S') \\ &\ge \frac{1}{1+\epsilon}v(T)-p(T)-(\alpha-1+2\epsilon+3\alpha\epsilon)v(S), \end{aligned}$$

where the last inequality uses v's ϵ-closeness to r. Thus $\mu\ge((1+\epsilon)(\alpha+2\epsilon+3\alpha\epsilon))^{-1}\ge(\alpha(1+3\epsilon)^2)^{-1}$. This completes the proof of Lemma 29. ◀

Proof of Theorem 21. The proof follows directly from Proposition 23 combined with Lemma 29. ◀

C.1 Proof of Claim 31

Towards proving Claim 31, the following is a property of valuations that are α-submodular and ϵ-close to unweighted transversal valuations. It can be seen as a strengthening of Observation 9.

▶ **Claim 32.** *For every part P and bundles $X\subseteq P$ and Y, if P is represented in Y then $v(X\mid Y)\le\alpha\epsilon$.*

Proof. Let y be an item representing P in Y. Then by α-submodularity, $v(X\mid Y)\le\alpha v(X\mid y)\le\alpha r(X\mid y)+\alpha\epsilon r(X\cup\{y\})=\alpha\epsilon$, where the last inequality is by invoking Observation 9. ◀

Proof of Claim 31. Fix a part P that is represented in S' and T'. Let t be an item representing P in T'. Order the items representing P in S' according to the order in which the Kelso-Crawford algorithm added them to the player's allocation for the last time (i.e., at their termination prices according to p), breaking ties arbitrarily; denote the ordered set by $(s_1,s_2,\dots)$. Let B_j be the bundle of the player right before item s_j was added, and let D_j be the set of items (not including s_j) with which s_j was added to B_j.

We first consider item s_1. If it is the case that P is not represented in B_1 and no other item represents P in D_1, then we argue that $v(s_1\mid B_1\cup D_1)-p(s_1)\ge v(t\mid B_1\cup D_1)-p(t)$. The reason for this is that otherwise, the player's utility could have been improved by replacing s_1 by t (using that t's termination price $p(t)$ is weakly higher than its price when s_1 was added). By monotonicity, we can write $v(s_1\mid B_1\cup D_1)\le v(s_1,t\mid B_1\cup D_1)\le v(t\mid B_1\cup D_1)+\alpha\epsilon$, where the last inequality is by Claim 32. Therefore $p(s_1)\le p(t)+\alpha\epsilon$. In the remaining case, P is represented in either B_1 or D_1. We know that $v(s_1\mid B_1\cup D_1)\ge p(s_1)$, otherwise the utility could have been improved by dropping s_1. By Claim 32, the left-hand side is at most $\alpha\epsilon$, and so $p(s_1)\le\alpha\epsilon$.

Now consider the rest of the items $s_2,s_3,\dots$. When item s_j is added, $\{s_1,\dots,s_{j-1}\}\subseteq B_j\cup D_j$. We also know that, as above, $v(s_j\mid B_j\cup D_j)\ge p(s_j)$. By α-submodularity this means that $\alpha\sum_{j\ge2}v(s_j\mid S_j)\ge\sum_{j\ge2}p(s_j)$. The left-hand side is equal to $\alpha v(\{s_2,s_3,\dots\}\mid s_1)$, and is therefore $\le\alpha\epsilon$ by Observation 9. We have thus shown that $\sum_{j\ge2}p(s_j)\le\alpha\epsilon$, and the same argument shows that the total payment for items in $S'\cap P$ that do not represent P is at most $\alpha\epsilon$.

We conclude that the total payment for items in $S'\cap P$ is at most $p(t)+3\alpha\epsilon$. Summing up over all parts represented in S' and T' completes the proof of the claim. ◀

D Negative Results for Specific Algorithms

How well do standard algorithms for welfare maximization perform for valuations that are close to, but are not quite, gross substitutes? After discussing the LP-based approach in Section 3.2, in this section we discuss the two other main approaches to welfare maximization for gross substitutes – the ascending auction algorithm of Kelso and Crawford [24], and the cycle canceling algorithm of Murota [31, 32]. Some of the details are deferred to the full version [40] due to space limitations.

At first glance, the Kelso-Crawford algorithm seems like a promising approach due to its known welfare guarantee of a 1/2-approximation for the class of submodular valuations [17]. However Proposition 20 and its proof in the full version [40, Proposition 8] show that the Kelso-Crawford algorithm cannot in general guarantee much better than that, even for simple submodular valuations that are arbitrarily close to unit-demand. More precisely, for every $\epsilon \leq 1$ there exists a market with $O(1/\epsilon)$ players whose submodular valuations are ϵ-close to unit-demand, and for which the Kelso-Crawford algorithm with adversarial ordering of the players finds an allocation with $\approx 2/3$ of the optimal welfare. An interesting open question is whether the Kelso-Crawford algorithm can be modified to eliminate such bad examples, e.g., by ordering the players in an optimal way.

The cycle canceling algorithm of Murota is a different approach which relies on properties of GS (or equivalently $M^\natural$-concave) valuations under local improvements. In Section D.1 we show that such properties hold for submodular valuations in a certain approximate sense, but unfortunately the cycle canceling algorithm cannot find a local optimum that would allow us to exploit these properties. In fact we show that a local optimum in Murota's sense does not exist for submodular valuations.

D.1 Murota's Cycle Canceling Approach

The cycle canceling approach is based on the following useful property of gross substitutes valuations, called the *single improvement (SI)* property [18]: Given a price vector p, we say that a bundle S is in *local demand* for a valuation v if its utility cannot be improved by adding an item, removing an item or swapping one item for another. The SI property holds if every bundle S in local demand is also in (global) demand (i.e., $v(S) - p(S) \geq v(T) - p(T)$ for every bundle T). Murota's cycle canceling algorithm finds an allocation and prices such that each player gets a bundle in local demand, thus arriving at an optimal allocation for gross substitutes valuations.

We observe that a variant of the SI property characterizes submodular valuations, and thus the SI property interpolates between submodularity and gross substitutes.

▶ **Definition 33.** Let $\beta \in [0, 1]$. A valuation v is β-SI if for every S in local demand, S is strongly individually rational,[14] and $v(S) - \beta p(S) \geq v(T) - p(T)$ for every T.

▶ **Observation 34.** *A monotone valuation is submodular if and only if it is* 0*-SI. A monotone valuation is gross substitutes if and only if it is* 1*-SI.*

Proof. Suppose v is submodular; we want to prove that v is 0-SI. Let S be in local demand, $v(S+i) - v(S) \leq p_i$, and $v(S-j) - v(S) \leq -p_j$. (We don't even need the swap property.) By submodularity and the second property, we have $v(S \cup T) \leq v(S) + \sum_{i \in T \setminus S} p_i$. Therefore,

[14] Being strongly IR implicitly holds for SI as well, i.e., if v is SI and S is in local demand then S is strongly IR.

$v(S) \geq v(S \cup T) - p(T \setminus S) \geq v(T) - p(T)$ by monotonicity. Also, for every $S' \subset S$, by submodularity we have $v(S \setminus S') \leq v(S) - \sum_{j \in S'} p_j$. Therefore $v(S') \geq v(S) - v(S \setminus S') \geq p(S')$.

Conversely, suppose that v is not submodular, i.e. $v(S+a+b) > v(S) + v(a \mid S) + v(b \mid S)$ for some S and $a, b \notin S$. We set $p_i = 0$ for $i \in S$ and $p_j = v(j \mid S)$ for $j \notin S$. Clearly S is in local demand, because swapping at most 1 element does not increase utility. However, $v(S \cup \{a, b\}) - p(S \cup \{a, b\}) > v(S)$, so v is not 0-SI. ◀

Moreover, as the next observation shows, finding an allocation with prices such that each player gets a bundle in their local demand would give a smooth transition would provide a smooth transition between a 1/2-approximation for submodular valuations and an optimal solution for gross substitutes.

▶ **Observation 35.** *For β-SI valuations, any allocation (of all items) with prices such that each player gets a bundle in their local demand has value at least $\frac{1}{2-\beta}\,\mathrm{OPT}$.*

Proof. Suppose that each v_i is β-SI and we have an allocation $(A_1, \ldots, A_n)$ with prices such that each A_i is in the local demand of player i. We assume that all items are allocated, and the allocation is individually rational; therefore, $\sum_{i=1}^n v_i(A_i) \geq \sum_{i=1}^n p(A_i) = \sum p_j$.

Suppose that the optimal allocation is $(O_1, \ldots, O_n)$. By the β-SI property, we have $v_i(A_i) - \beta p(A_i) \geq v_i(O_i) - p(O_i)$. Adding up over all players, $\sum_{i=1}^n v_i(A_i) - \beta \sum p_j \geq \mathrm{OPT} - \sum p_j$. From here, $\sum_{i=1}^n v_i(A_i) \geq \mathrm{OPT} - (1-\beta) \sum p_j \geq \mathrm{OPT} - (1-\beta) \sum_{i=1}^n v_i(A_i)$. Thus $(2-\beta) \sum_{i=1}^n v_i(A_i) \geq \mathrm{OPT}$. ◀

Unfortunately, this approach (whether by cycle canceling or by any other method) is doomed to fail: Example 1 in the full version [40] shows a market with submodular valuations for which it is impossible to find an allocation and prices such that every player's bundle is in their local demand. In other words, for this market it is impossible to get rid of all negative cycles in Murota's algorithm, ruling out this approach.

Glauber Dynamics for Ising Model on Convergent Dense Graph Sequences*

Rupam Acharyya[1] and Daniel Štefankovič[2]

1 Department of Computer Science, University of Rochester, Rochester, NY, USA
racharyy@cs.rochester.edu

2 Department of Computer Science, University of Rochester, Rochester, NY, USA
stefanko@cs.rochester.edu

Abstract

We study the Glauber dynamics for Ising model on (sequences of) dense graphs. We view the dense graphs through the lens of graphons [19]. For the ferromagnetic Ising model with inverse temperature β on a convergent sequence of graphs $\{G_n\}$ with limit graphon W we show fast mixing of the Glauber dynamics if $\beta\lambda_1(W) < 1$ and slow (torpid) mixing if $\beta\lambda_1(W) > 1$ (where $\lambda_1(W)$ is the largest eigenvalue of the graphon). We also show that in the case $\beta\lambda_1(W) = 1$ there is insufficient information to determine the mixing time (it can be either fast or slow).

1998 ACM Subject Classification G.3 Probability and Statistics

Keywords and phrases Spin systems, Glauber dynamics, Ising model, graphons

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.23

1 Introduction

Spin systems have been extensively studied in physics [11], mathematics [25], and machine learning [24]. An important and challenging computational question is efficiently sampling configurations from the distribution of a model (spin system). A popular sampling method (and the focus of our paper) is *Glauber dynamics* [11]. One of the most studied spin models is *Ising model* [14, 12]. Even though there is a polynomial-time algorithm to sample from the distribution of the ferromagnetic Ising model [13] it is still useful (for reasons of simplicity, generality, and speed) to study the Glauber dynamics for the model [15, 21]. A basic question is: what properties of the underlying graph and the temperature make the Glauber dynamics fast (or slow)? In the case of sparse graphs the dynamics was studied for, for example, $\mathbb{Z}^2$ (see, e.g., [20]), general bounded degree graphs [22], and graphs with bounded connective constant [27, 26].

In the case of dense graphs the dynamics was studied for the complete graph [15] (for more general models on the complete graph, see [6, 2]). Our goal is to understand the impact of the structural properties (analaogously to the connective constant) of the dense graphs and the speed of Glauber dynamics. We will view dense graphs through the lens of *graphons* [19] and use the notions of free energy of a spin system on a graphon [4]. We give a threshold for the inverse temperature below which Glauber dynamics is rapidly mixing

* This material is based upon work supported by the National Science Foundation under NSF grant CCF-1563757.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 23; pp. 23:1–23:22
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

and above which the mixing is slow. This generalizes [15] from complete graphs to general dense graph sequences. We also show that at the critical point it is not possible to draw a conclusion about the mixing time for a convergent sequence of graphs just by looking at the limit graphon.

We obtain our lower bound results by studying the typical configurations of the model [23, 15]. A phase of a spin configuration denotes what fraction of vertices get what spin. The most probable (dominant) phases play an important role in influencing the speed of the Glauber dynamics. Intuitively, a unique dominant phase (at a high temperature) corresponds to fast mixing of Glauber dynamics, whereas multiple dominant phases (at a low temperature) correspond to slow mixing of Glauber (and other) dynamics (moving between phases requires the chain to move through a high energy barrier). The typical phases were previously studied, for example, to show slow mixing of Glauber dynamics [23, 15, 8] and to prove hardness results for sampling [10, 9, 28].

2 Background

2.1 Homogeneous Ferromagnetic Ising Model (with no external field)

Ising Model was introduced in 1920's by Lenz [14] and Ising [12]. Let $G = (V(G), E(G))$ be a finite graph. In a configuration of the model, each vertex is assigned a spin from the set $\{+1, -1\}$. The energy of a configuration σ, is specified by the Hamiltonian of the configuration

$$H(\sigma) = -\sum_{v\sim w} J(v,w)\sigma(v)\sigma(w),$$

where $v \sim w$ denotes v is a neighbor of w in G and $J(v,w)$ denotes the interaction strength between vertices v and w.

We study homogeneous ferromagnetic Ising Model, that is, we assume $J(v,w) = 1$ for all $v, w \in V$. The probability measure μ, on the set of configurations $\Omega = \{+1, -1\}^{|V(G)|}$, for this model is given by,

$$\mu(\sigma) = \frac{e^{-\beta H(\sigma)}}{Z(\beta)},$$

where $\beta > 0$ is called the inverse temperature and $Z(\beta)$, the normalization factor is called the *partition function*.

This work focuses on dense graphs. We follow [15] in re-parameterizing the inverse temperature β as β/n, where $n = |V(G)|$. So the probability measure for dense graphs can be rewritten as,

$$\mu(\sigma) = \frac{e^{(\beta/n)\cdot S(\sigma)}}{Z(\beta)},$$

where $S(\sigma) = \sum_{v,w\in V, v\sim w} \sigma(v)\sigma(w)$.

2.2 Glauber Dynamics

In this paper we analyze Glauber Dynamics to sample from the distribution of the model. The (single site) Glauber Dynamics for the probability measure μ is defined by the following transition rule.

1. Pick a vertex v (also called site) uniformly at random from $V(G)$.
2. Change the spin of v with respect to the spins of its neighbors, i.e., in the new configuration, spin of v will be $+1$ with a probability of $p(\sigma, v)$, where

$$p(\sigma, v) := \frac{\exp(\frac{\beta}{n} S_v(\sigma))}{\exp(\frac{\beta}{n} S_v(\sigma)) + \exp(-\frac{\beta}{n} S_v(\sigma))},$$

and $S_v(\sigma) = \sum_{w \in V, v \sim w} \sigma(w)$.

We study the following (standard) notion of mixing time. The mixing time $\tau_{mix}(\varepsilon)$ of a Markov chain with state space Ω, transition matrix P and stationary distribution π is

$$\tau_{mix}(\varepsilon) = \max_{X_0 \in \Omega} \min\{t : d_{TV}(P^t(X_0, \cdot), \pi) \le \varepsilon\}.$$

Usually $\varepsilon = \frac{1}{4}$ or $\varepsilon = \frac{1}{2e}$ is used.

2.3 Convergent Sequence of Dense Graphs

We study sequences of dense graphs using notions of convergence defined in [3, 4].

Let G be a weighted graph with non-negative vertex weights α_v that sum to 1 and edge weights $\beta_{uv} \in [0, 1]$. Let G' be another weighted graph with non-negative vertex weights α'_i that sum to 1 and edge weights $\beta'_{ij} \in [0, 1]$. Let $\chi(G, G')$ be the set of fractional overlays between G and G', where a fractional overlay (between G and G') is $X \in \mathbb{R}_{\ge 0}^{V(G) \times V(G')}$ such that $\sum_i X_{vi} = \alpha_v(G)$ and $\sum_v X_{vi} = \alpha'_i(G')$. The *cut distance* between G and G' is (see [19])

$$\delta_\square(G, G') = \min_{X \in \chi(G, G')} d_\square(G, G', X), \tag{1}$$

where

$$d_\square(G_n, G, X) = \max_{Q, R \subset V(G) \times V(G')} \left| \sum_{(v,i) \in Q, (u,j) \in R} X_{vi} X_{uj} (\beta_{uv} - \beta'_{ij}) \right|. \tag{2}$$

The *free energy* of an Ising model with parameter β/n for a dense graph G_n is defined as follows (see [17]).

$$\hat{\mathcal{F}}(G_n, \beta) = -\frac{1}{|V(G_n)|} \ln Z(G_n, \beta),$$

where $Z(G_n, \beta) = \sum_{\sigma: V(G_n) \to \{+1, -1\}} \exp(\frac{1}{n} \sum_{(u,v) \in E(G_n)} \beta \sigma(u) \sigma(v))$.

Microcanonical free energy is a more detailed version of free energy – we compute the free energy for each phase (by phase we mean the fraction of the vertices with positive spin), formally defined for $a \in [0, 1]$ as follows (see [17]):

$$\hat{\mathcal{F}}_a(G, \beta) = -\frac{1}{n} \ln Z_a(G, \beta),$$

where $Z_a(G, \beta) = \sum_{\sigma \in \Omega_a(G)} \exp(\frac{\beta}{n} \sum_{(u,v) \in E(G)} \sigma(u) \sigma(v))$ and

$$\Omega_a(G) = \{\sigma : V(G) \to \{-1, +1\} \,\big|\, \big||\sigma^{-1}(\{+1\})| - a|V(G)|\big| \le 1\}.$$

In [4] it has been shown that convergence w.r.t. cut metric implies convergence w.r.t. microcanonical free energy and free energy (they also show converse if one has convergence w.r.t. microcanonical free energies for all spin models).

2.4 Limit Object of Convergence: Graphon

The limits of the convergence w.r.t. the cut norm are graphons [19].

▶ **Definition 1** (Graphon, [19])**.** A graphon W is a symmetric measurable function $W : [0,1]^2 \to [0,1]$. (The symmetry means $W(x,y) = W(y,x)$ for all $x, y \in [0,1]$.)

The simplest graphons correspond to step functions with finitely many steps.

▶ **Definition 2** (Step Graphon, [19])**.** Let $S_1, \dots, S_k$ be a disjoint decomposition of $[0,1]$ into intervals for some finite k and let P be a symmetric $k \times k$ matrix with entries from $[0,1]$. A function $U : [0,1]^2 \to [0,1]$ is a step graphon with value matrix P if $\forall i, j$ and $\forall (x,y) \in S_i \times S_j$ $U(x,y) = P_{ij}$. We call $\alpha_1, \dots, \alpha_k$ the step sizes of the step graphon, where $\alpha_i = |S_i|$ for $i \in \{1, \dots, k\}$.

Given a weighted graph H (with $|V(H)| = n$) a step graphon W_H can be naturally constructed as follows. Let $S_1, \dots, S_n$ be disjoint sub-intervals of $[0,1]$ such that S_i is of size α_i, where α_i is the weight of the vertex $i \in V(G)$. For $x \in S_i$ and $y \in S_j$ we let $W_H(x,y) = \beta_{ij}$, where β_{ij} is the weight of the edge between vertices i and j (if there is no edge between i and j we let $\beta_{ij} = 0$).

▶ **Definition 3** (Eigenvalue of a Graphon)**.** [17] Given a graphon W, consider the following operator $T_W : L_2[0,1] \to L_1[0,1]$:

$$(T_W f)(x) = \int_{[0,1]} W(x,y) f(y)\, dy.$$

The operator T_W has discrete spectrum, i.e., a multi-set of real nonzero eigenvalues $\lambda_1, \lambda_2, \dots$ (sorted in the non-increasing order by their absolute value), such that $\lambda_n \to 0$. We call these the eigenvalues of the graphon W. The eigenvalue with highest absolute value is denoted $\lambda_1(W)$.

The notions of cut distance, free energy, and micro-canonical free energy extend from graphs to graphons (see [17]).

The *cut distance* between two graphons is:

$$\delta_\Box(W,U) := \inf_\phi \|W^\phi - U\|_\Box := \inf_\phi \sup_{S,T} \left| \int_{S\times T} W^\phi(x,y) - U(x,y)\, dx\, dy \right|,$$

where $\phi : [0,1] \to [0,1]$ is a measure preserving function and $W^\phi(x,y) = W(\phi(x), \phi(y))$. The cut distance between a graph G and a step graphon W is denoted by $\delta_\Box(G,W) = \delta_\Box(W_G, W)$.

The *free energy* of a graphon is defined as

$$\mathcal{F}(W,\beta) = \inf_{m:[0,1]\to[-1,1]} \mathcal{E}(W,\beta,m), \tag{3}$$

where

$$\mathcal{E}(W,\beta,m) = -\frac{\beta}{2}\langle m, T_W m\rangle - \mathrm{Ent}(m), \tag{4}$$

and

$$\mathrm{Ent}(m) = -\int_0^1 \frac{1}{2}(1-m(x)) \log(\frac{1}{2}(1-m(x)))\, dx - \int_0^1 \frac{1}{2}(1+m(x)) \log(\frac{1}{2}(1+m(x)))\, dx,$$

and

$$\langle m, T_W m\rangle = \int_{[0,1]^2} W(x,y)m(x)m(y)\,dx\,dy.$$

The *microcanonical free energy* of a graphon with phase $a \in [0,1]$ is

$$\mathcal{F}_a(W,\beta) = \inf_{m:[0,1]\to[-1,1] \text{ and } \int_{[0,1]} m(x)\,dx = a} \mathcal{E}(W,\beta,m), \tag{5}$$

where $\mathcal{E}(W,\beta,m)$ is defined as in (4).

A sequence of dense graphs $\{G_n\}$ is said to be convergent to a graphon W if $\delta_\square(G_n,W) \to 0$. For a sequence of dense graphs $\{G_n\}$ converging to a graphon W it has been shown [3, 4] that the free energy and microcanonical free energy of the dense graphs converge to the free energy and microcanonical free energy of the graphon.

▶ **Proposition 4** ([4]). *Suppose $\{G_n\}$ be a sequence of dense graphs convergent to a graphon $W : [0,1]^2 \to [0,1]$. Then*
1. $\hat{\mathcal{F}}(G_n,\beta) \to \mathcal{F}(W,\beta)$.
2. $\forall a \in [0,1]$, $\hat{\mathcal{F}}_a(G_n,\beta) \to \mathcal{F}_a(W,\beta)$.

3 Main Results and Related Works

3.1 Results for Mixing Time

The Glauber dynamics for Ising model has been extensively studied in [13, 15, 22]. The dynamics is well understood when the graph has bounded degree [13, 21, 1]. In the dense scenario the dynamics has been analyzed for the complete graph [15] (so-called mean field model). The complete graph corresponds to graphon with $W(x,y) = 1$ (for all $x, y \in [0,1]$). Our goal here is to extend this work to general graphons (we aim to understand the connection between the mixing time, the inverse temperature, and the structure of the graphon).

For the Ising model on the complete graph [15] show that the mixing of Glauber Dynamics is fast when $\beta < 1$ and it is exponentially slow when $\beta > 1$. This threshold behavior extends to convergent dense graph sequences – we provide a threshold for the parameter β, such that, below the threshold mixing of Glauber dynamics is fast and above the threshold mixing is slow (our result matches the threshold for the complete graph – the threshold is $\beta = 1/\lambda_1(W)$ and for complete graph $\lambda(W) = 1$). Formally we have the following results.

▶ **Theorem 5.** *Consider a homogeneous ferromagnetic Ising model (with no external field) with inverse temperature β and a graphon W. If $\{G_n\} \to W$, then the mixing time of the Glauber Dynamics for Ising model on G_n satisfies the following:*
1. *If $\lambda_1(W) \cdot \beta < 1$ then $\tau_{mix}(G_n) = O(n \log(n))$.*
2. *If $\lambda_1(W) \cdot \beta > 1$ then $\tau_{mix}(G_n) = e^{\Omega(n)}$.*

Remark (mixing in critical case): In the above theorem we haven't stated any result for the critical temperature, i.e., when $\lambda_1(W)\beta = 1$. This is because, at the critical temperature one cannot draw conclusion about the mixing time for a convergent sequence of graphs just by looking at the limit graphon. We show examples of two different graph sequences which converge to the same graphon, even though at critical temperature mixing is fast for one sequence and slow for the other. These examples are discussed in Section 9.

3.2 Results for Phase Diagram

A phase α of the Ising model is the set of configurations which has αn fraction of vertices with $+1$ spin. The weight of a phase is the value of the partition function when restricted to the configurations with the given phase signature. The phase which has maximum weight is called the *dominant phase*. It has been seen earlier that when the model is studied on a graph, the phase diagram of the model changes with different values of the parameter β. For example, when Ising model is studied on complete graphs the model exhibits an unique dominant phase if $\beta < 1$ and it has multiple dominant phases when $\beta > 1$. It has been shown in [15] that coexistence of multiple dominant phases implies slow mixing, because to get from one phase to another it requires to pass through a high free energy barrier. Hence studying phase diagram for spin models has been focus of numerous previous studies [23, 10, 9]. The goal of these studies was to understand the speed of the dynamics. As we know from Section 2.4 that the free energy is defined as the negative of the logarithm of the partition function, to find the dominant phase we need to find the phase which minimizes the free energy. In this paper our interest is to study the behavior of the phase transition on a sequence of graphs. For this purpose we study the behavior of the free energy on the limit graphon, i.e., we try to find for what values of β there is an unique minimizer (equivalently unique dominant phase) in the expression for the free energy. Formally we have the following theorem.

▶ **Theorem 6.** *Consider a graphon W and the free energy function for the graphon W with respect to the inverse temperature parameter β is defined as in* (4)*.*

1. *If $\lambda_1(W) \cdot \beta < 1$ then the function $\mathcal{E}(W, \beta, m)$ has unique*[1] *local minimum.*
2. *If $\lambda_1(W) \cdot \beta > 1$ then the function $\mathcal{E}(W, \beta, m)$ has multiple*[2] *global minima.*

4 Organization

In Section 5 we prove for a convergent graph sequence than one can align the graphs in the sequence with a step graphon (that is close to the limit graphon) in such a way that most vertices have same neighborhood statistics as the step graphon. This property will later be used to prove the upper bound result of Theorem 5. Next in Section 6 we establish the phase digram for different values of β (Theorem 6). The result about phase diagram is an important tool to prove the lower bound of mixing time of Theorem 5. Finally in Section 7 we prove the upper bound result at high temperature and in Section 8 we prove that the mixing is slow on the graphs in the sequence at low temperature. All the remaining proofs can be found in the Appendix.

5 Labeling Graphs in a Convergent Graph Sequence

In this section we will deduce some properties of convergent graph sequences which will be used to prove the upper bound result of Theorem 5.

▶ **Definition 7** (GOOD and BAD vertices)**.** Let U be a step graphon with k steps, value matrix P, and step sizes $\alpha_1, \ldots, \alpha_k$. Let G be a graph and let $\phi : V(G) \to \{1, \ldots, k\}$ be a

[1] By unique we mean unique up to measurability, i.e., m_1 and m_2 are two solutions then the set where they differ has measure zero

[2] By multiple we mean there exists at least two functions m_1 and m_2 such that the set where they differ has measure greater than zero

labeling. Let v be a vertex of G and let $i = \phi(v)$. We call the vertex v to be GOOD with ε tolerance if for all $j \in \{1, \ldots, k\}$,

$$|\{w \,|\, w \sim v; \phi(w) = j\}| \leq (P_{ij}\alpha_j + \varepsilon)n.$$

Otherwise we call the vertex to be BAD w.r.t. ε tolerance.

▶ **Definition 8** (Proper Labeling). Let G be a graph and U be a step graphon. A labeling $\phi : V(G) \to \{1, \ldots, k\}$ is said to be proper up to ε tolerance w.r.t. U if there are at most εn many BAD vertices w.r.t. ε tolerance.

With the above definitions we can now state the following lemma.

▶ **Lemma 9.** *Let $\{G_n\}$ be a sequence of graphs such that $G_n \to W$ for some graphon W. Then for any $\varepsilon > 0$ there exists $k = k(\varepsilon), n_0 = n_0(\varepsilon)$ and a step graphon U with k steps such that $\delta_\square(W, U) \leq \varepsilon$ and such that $\forall n \geq n_0$ we have that G_n has a proper labeling up to ε tolerance w.r.t. U.*

To prove the Lemma 9 we will first prove an easier version of the lemma when the limit graphon is a step graphon.

▶ **Lemma 10.** *Let $\{G_n\}$ be a sequence of graphs such that $G_n \to U$ for some step graphon U. Then for any $\varepsilon > 0$ there exists $n_0 = n_0(\varepsilon)$ such that $\forall n \geq n_0$ we have that G_n has a proper labeling up to ε tolerance w.r.t. U.*

Proof of Lemma 10. We know that $G_n \to U$ implies that for given $\varepsilon > 0$ there exists n_0 such that $\forall n \geq n_0$,

$$\delta_\square(G_n, U) \leq \varepsilon^2/2. \tag{6}$$

Since every step graphon can be viewed as arising from a weighted graph G by the construction shown in in Section 2.3, we will, w.l.o.g., assume $U = W_G$. Hence $\delta_\square(G_n, U) = \delta_\square(G_n, W_G) = \delta_\square(G_n, G)$. Now for two weighted graphs we have

$$\delta_\square(G_n, G) = \min_{X \in \chi(G_n, G)} d_\square(G_n, G, X), \tag{7}$$

where X is a fractional overlay, i.e., $\sum_i X_{vi} = \frac{1}{n}$ and $\sum_v X_{vi} = \alpha_i(G)$ and

$$d_\square(G_n, G, X) = \max_{Q,R \subset V(G_n) \times V(G)} \left| \sum_{(v,i) \in Q, (u,j) \in R} X_{vi}X_{uj}(1 - P_{ij}) \right|. \tag{8}$$

Note that we give weight $\frac{1}{n}$ to each vertex $v \in V(G_n)$ (as G_n is originally unweighted). The 1 in (8) is the weight of the edge $(u, v) \in E(G_n)$. Similarly $\alpha_i(G)$ is the weight of the vertex $i \in V(G)$ and P_{ij} is the weight of the edge $(i, j) \in E(G)$. Now let X be the fractional overlay which minimizes the cut distance. We assign the label ϕ of a vertex $v \in V(G_n)$ from the distribution $\{nX_{vi}\}_i$, i.e., $\phi(v) = i$ with probability nX_{vi}. Note that for any vertex,

$$\mathbb{E}\Big[\big|\{w|w \sim v; \phi(w) = j\}\big|\Big] = n \sum_{w|w \sim v} X_{wj}.$$

Now we call a vertex v to be *dangerous* for (i, j) if $\phi(v) = i$ and

$$\sum_{w|w \sim v} X_{wj} \geq \alpha_j P_{ij} + \frac{\varepsilon}{2}. \tag{9}$$

Now we will show that there are not too many such dangerous vertices.

Bound on number of Dangerous Vertices: First we fix i and j. Let Q be the set of all dangerous vertices for (i,j) and R be the set of all vertices $w \in V(G)$ with label j . Then from (9), (6) and (8) we have:

$$\sum_{v\in Q} X_{vi}(\alpha_j P_{ij} + \frac{\varepsilon}{2}) \leq \sum_{v\in Q} X_{vi} \sum_{w\sim v} X_{wj} \leq \sum_{v\in Q} X_{vi} \sum_{w\in R} X_{wj} P_{ij} + \frac{\varepsilon^2}{2} = \sum_{v\in Q} X_{vi}\alpha_j P_{ij} + \frac{\varepsilon^2}{2}. \tag{10}$$

Hence from (10) we have $\sum_{v\in Q} X_{vi} \leq \varepsilon$. So from Chernoff Bound w.h.p. the number of dangerous vertices are at most εn. Next we look at the vertices which are not dangerous for any (i,j), i.e., if $\phi(v) = i$, then for all j we have

$$\sum_{w|w\sim v} X_{wj} \leq \alpha_j P_{ij} + \frac{\varepsilon}{2}. \tag{11}$$

We now move on to prove that the probability there there exists too many BAD vertices is very low. We now use Y_v as an indicator variable to denote whether the vertex v is BAD or not. Hence it is enough to bound $Pr[\sum_v Y_v \geq \varepsilon n]$. Now from Markov's inequality we have:

$$Pr[\sum_v Y_v \geq \varepsilon n] \leq \frac{\sum_v \mathbb{E}[Y_v]}{\varepsilon n}. \tag{12}$$

Hence we now need to bound $\mathbb{E}[Y_v]$. Again using Markov's inequality we have

$$\begin{aligned} \mathbb{E}[Y_v] &= Pr[v \text{ is BAD}] \\ &= \sum_i Pr[v \text{ gets label } i] \cdot Pr[\exists j \ni |\{w|w \sim v; \phi(w) = j\}| \geq (P_{ij}\alpha_j + \varepsilon)n \Big| \phi(v) = i] \\ &\leq \sum_i nX_{vi} \sum_j Pr[|\{w|w \sim v; \phi(w) = j\}| \geq (P_{ij}\alpha_j + \varepsilon)n]. \end{aligned} \tag{13}$$

Using Chernoff-Hoeffding bound for any non-dangerous vertex v we have,

$$Pr[|\{w|w \sim v; \phi(w) = j\}| \geq (P_{ij}\alpha_j + \varepsilon)n] \leq \exp(-n\varepsilon^2/4). \tag{14}$$

Now from (12) and (13) we have:

$$Pr[\sum_v Y_v \geq \varepsilon n] \leq \frac{kn\exp(-n\varepsilon^2/4)}{\varepsilon n} = \frac{k}{\varepsilon}\exp(-n\varepsilon^2/4).$$

Hence we have the lemma. ◀

Proof of Lemma 9. As $G_n \to W$ we have for given $\varepsilon > 0$ there exists n_0 such that

$$\delta_\square(G_n, W) \leq \frac{\varepsilon^2}{4}. \tag{15}$$

Also from [17] we have for any graphon W we have that $\exists$ a step function U with k steps (where k is sufficiently large) such that

$$\delta_\square(U, W) \leq \sqrt{\frac{2}{\log_2 k}}\|U\|_2 \leq \frac{\varepsilon^2}{4}. \tag{16}$$

Hence from (15) and (16) we have the following analog of (6)

$$\delta_\square(U, G_n) = \delta_\square(G_U, G_n) \leq \frac{\varepsilon^2}{2}, \tag{17}$$

where G_U is a graph on k vertices. Now the remainder of the proof of the lemma is identical to the proof of Lemma 10. ◀

6 Phase Diagram

In this section we will prove Theorem 6. As free energy of the model is the infimum over the set of all measurable functions from $[0,1]$ to $[-1,1]$ (defined in Section 2.4) we first need to prove that there exists some such function at which the infimum is achieved. Then we will analyze its properties.

▶ **Lemma 11.** *Let $\mathcal{E}(W,\beta,m)$ be the function as defined by* (4)*. Then the following infimum is attained for some measurable function m:*

$$\inf_{m:[0,1]\to[-1,1]} \mathcal{E}(W,\beta,m). \tag{18}$$

Proof of Lemma 11 has been deferred to Appendix. Assuming the existence we now move on to prove Theorem 6.

Proof of Theorem 6.

Case I: $\lambda_1(W)\beta < 1$. In this case we will prove that the functional $m \mapsto \mathcal{E}(W,\beta,m)$ is strictly convex. Then there will be an unique minimum up to measurability (strict convexity implies unique minimum because if there were two minima available then by strict convexity there average will have a strictly lesser functional value which is a contradiction). Formally we prove the following lemma.

▶ **Lemma 12.** *$\mathcal{E}(W,\beta,m)$ is defined as in* (3)*. Then for all $0 \le \alpha \le 1$ and for all measurable functions m,p from $[0,1]$ to $[-1,1]$ we have :*

$$\mathcal{E}(W,\beta,(1-\alpha)m+\alpha p) < (1-\alpha)\mathcal{E}(W,\beta,m)+\alpha\mathcal{E}(W,\beta,p).$$

whenever $\lambda_1(W)\beta < 1$.

We prove the above lemma in the Appendix.

Case II: $\lambda_1(W)\beta > 1$. For the purpose of the proof we slightly re-parameterize the functions. In particular we define $\rho(x) := \frac{1}{2}(m(x)+1)$. Hence the optimization problem can be written as:

$$\inf_{\rho:[0,1]\to[0,1]} \mathcal{E}(W,\beta,2\rho-1). \tag{19}$$

If two measurable functions $f,g : [0,1] \to [0,1]$ differ on a set of measure zero we write $f \underset{m}{\approx} g$. Now we define a new set $S = \{\rho : [0,1] \to [0,1] | \int_{[0,1]} \rho(x)\,dx = \frac{1}{2}\}$. We will show that the minimum doesn't lie in the set S. For the function $\rho(x) = 1/2$ everywhere we argue that it cannot be the minimum by a local perturbation argument. For all the other functions $\rho \in S$ we use the following transformation to produce a function with a smaller value.

▶ **Definition 13.** Given a function $\rho \in S$ we define another measurable function $\hat{\rho} : [0,1] \to [0,1]$ as follows:

$$\hat{\rho}(x) = \begin{cases} \rho(x) & \textit{if } \rho(x) \ge \frac{1}{2}, \\ 1-\rho(x) & \textit{otherwise.} \end{cases}$$

Now we have the following lemma for $\hat{\rho}$ the proof of which has been deferred to Appendix.

▶ **Lemma 14.** *If $\rho \in S = \{\rho : [0,1] \to [0,1] | \int_{[0,1]} \rho(x)\,dx = \frac{1}{2}\} \setminus \{\rho : [0,1] \to [0,1] | f \underset{m}{\approx} \rho$ and $\rho(x) = \frac{1}{2} \forall x \in [0,1]\}$ and $\hat{\rho}$ is defined as in Definition 13, then*

$$\mathcal{E}(W, \beta, 2\hat{\rho} - 1) < \mathcal{E}(W, \beta, 2\rho - 1). \tag{20}$$

It remains to rule out the function $\rho(x) = \frac{1}{2}$ (for $x \in [0,1]$), that is, to show that this function is also not an minimum point for $\mathcal{E}(W, \beta, 2\rho - 1)$. More formally we have the following lemma.

▶ **Lemma 15.** *Consider the following minimization problem from* (18)

$$\inf_{\rho:[0,1]\to[0,1]} \mathcal{E}(W, \beta, 2\rho - 1).$$

If $\lambda_1(W)\beta > 1$ then $\rho : [0,1] \to [0,1] | \rho(x) = \frac{1}{2} \forall x$ is not a minimizer of the optimization problem.

Hence from the Lemma 14 and 15 we have that the minimizers of $\mathcal{E}(W, \beta, 2\rho-1)$ is not in the set $S = \{\rho : [0,1] \to [0,1] | \int_{[0,1]} \rho(x)\,dx = \frac{1}{2}\}$. Note that $\mathcal{E}(W, \beta, 2\rho-1) = \mathcal{E}(W, \beta, 2(1-\rho)-1)$. Hence if ρ_{opt} is a minimizer of $\mathcal{E}(W, \beta, 2\rho - 1)$ so is $1 - \rho_{opt}$. Hence the optimization problem has multiple minima. ◀

7 Upper Bound for the Mixing Time

We will now prove the upper bound result stated in the Theorem 5 using path coupling, a well known proof technique for bounding mixing time. We state a lemma from [5] which will be used for the proof.

▶ **Lemma 16.** *[5] Let $\mathcal{X}$ be a Markov chain. Let $G_{\mathcal{X}}$ be the graph of the Markov chain. Let ℓ be a length function on the edges of $G_{\mathcal{X}}$ such that $\ell(x,y) \geq 1$ for each edge $\{x,y\} \in E(G_{\mathcal{X}})$. This then naturally extends to a metric (which we also denote by ℓ), where $\ell(x,y)$ is the length of the shortest path from x to y. Suppose that for each edge $(x,y) \in G_{\mathcal{X}}$ there exists a coupling (X_1, Y_1) of $P(x,\cdot)$ and $P(y,\cdot)$ such that the following holds:*

$$\mathbb{E}_{x,y}[\ell(X_1, Y_1)] \leq \ell(x,y) e^{-\alpha}.$$

Then

$$t_{mix}(\eta) \leq \left\lceil \frac{-\log(\eta) + \log(\operatorname{diam}(\mathcal{X}))}{\alpha} \right\rceil,$$

where $\operatorname{diam}(\mathcal{X}) = \max_{x,y \in G_{\mathcal{X}}} \ell(x,y)$ *is the diameter of* $G_{\mathcal{X}}$

Now we prove the main theorem about fast mixing in high temperature.

Proof of Theorem 5.1. From Lemma 9 we know that for any $\varepsilon > 0$ for any sufficiently large n the graph G_n can be properly labeled up to ε tolerance (call) w.r.t. some step graphon U such that U is ε-close to the limit graphon W, i.e., $\delta_\square(U, W)\varepsilon$. Let's call the labeling as ϕ. Let k be the number of steps in U and $\alpha_1, \ldots, \alpha_k$ be the step sizes. Now we define the length function ℓ to be used in the path coupling argument.

Defining the Distance: For a vertex $v \in V(G)$ we define the following quantity,

$$\hat{d}_v = \begin{cases} d_{\phi(v)} & \text{if } v \text{ is GOOD w.r.t. } \phi, \\ \frac{1}{\lambda_1(U)} \sum_j d_j & \text{if } v \text{ is BAD w.r.t. } \phi, \end{cases}$$

where $(d_1, \ldots, d_k)$ is the eigenvector corresponding to the largest eigenvalue $(\lambda_1(U))$ of the step graphon U, where the eigenvector is scaled so that $d_i \geq 1$. Note that if for all i we have $d_i \geq 1$ then $\frac{1}{\lambda_1(U)} \sum_j d_j \geq 1$. Now the distance between any two arbitrary configurations σ' and τ' is defined as:

$$\ell(\sigma', \tau') = \sum_{v \in V(G) \text{ and } \sigma'(v) \neq \tau'(v)} \hat{d}_v.$$

Choice of ε: Now let $\varepsilon_0 > 0$ be such that $(\lambda_1(W) + \varepsilon_0)\beta = 1$. We will choose U such that $|\lambda_1(W) - \lambda_1(U)| \leq \varepsilon_0/4$ and take $\varepsilon > 0$ such that $\varepsilon d_{\text{bad}}(1 + \lambda_1(U)) = (\min_j d_j)\frac{\varepsilon_0}{4}$, where $d_{\text{bad}} = \frac{1}{\lambda_1(U)} \sum_j d_j$.

Defining the Path Coupling: Let σ, τ be two configurations such that the two configurations differ only at v and $\sigma(v) = -1$ and $\tau(v) = +1$. Now we describe a coupling (X, Y) such that X starts with σ and Y starts with τ.

- Pick one vertex w u.a.r from V.
- If $w \notin \mathcal{N}(v)$ then update the spin of w in both X and Y with transition probability specified by the dynamics [in Section 2.2].
- If $w \in \mathcal{N}(v)$ then pick a number $Z \in [0, 1]$ and set

  $$X_1(w) = \begin{cases} +1, & \text{if } Z \leq p(\sigma, v), \\ -1, & \text{otherwise,} \end{cases}$$

  and

  $$Y_1(w) = \begin{cases} +1, & \text{if } Z \leq p(\tau, v) \\ -1, & \text{otherwise,} \end{cases}$$

  where

  $$p(\sigma, v) = \frac{e^{\beta S_v(\sigma)}}{e^{\beta S_v(\sigma)} + e^{-\beta S_v(\sigma)}}, \tag{21}$$

  and $S_v(\sigma) = \sum_{v \sim w} \sigma(w)$.

From the definition of the coupling we can see that the disagreement of the two configurations spreads further with probability $p(\tau, v) - p(\sigma, v)$. We have the following upper bound on the probability of spreading disagreement.

▶ **Claim 17** (see, e.g., [16]). *Consider Ising model on a dense graph G with inverse temperature β and let σ, τ be two configurations such that the two configurations differ only at v and $\sigma(v) = -1$ and $\tau(v) = +1$. Also $p(\sigma, v)$ is defined as in (21). Then we have*

$$p(\tau, v) - p(\sigma, v) \leq \tanh(\frac{\beta}{n}).$$

Now we analyze the expected decrease of the coupling distance in two cases to satisfy the hypothesis of the Lemma 16.

Case I: v is GOOD: As we can see from Lemma 9 if v is a GOOD vertex then we have number of neighboring vertices of v with label j is $\le (P_{ij}\alpha_j + \varepsilon)n$. As we have seen in the coupling we choose a vertex w u.a.r., i.e., w.p. $\frac{1}{n}$. Now we have the following cases:

- If $w = v$, then $d(X_1, Y_1) = 0$.
- If $w \notin \mathcal{N}(v) \cup \{v\}$, then $d(X_1, Y_1) = d_i$.
- If $w \in \mathcal{N}(v)$ and w gets label j by the labeling, then w.p. $p(\tau, v) - p(\sigma, v)$,
  - $\ell(X_1, Y_1) = d_i + d_j$, if w is GOOD,
  - $\ell(X_1, Y_1) = d_i + d_{\text{bad}}$, if w is BAD.

where $d_{\text{bad}} = \frac{1}{\lambda_1(U)} \sum_j d_j$. Also from Lemma 9 there are at most εn many BAD vertices. So from Claim 17 and the above discussion we have

$$\begin{aligned}
\mathbb{E}[\ell(X_1, Y_1)] &\le d_i(1 - \frac{1}{n}) + \frac{1}{n} \cdot \tanh(\beta/n)\Big[\sum_j (P_{ij}\alpha_j + \varepsilon)n \cdot d_j + \varepsilon n \cdot d_{\text{bad}}\Big] \\
&\le d_i(1 - \frac{1}{n}) + \frac{1}{n} \cdot \beta\Big[\sum_j (P_{ij}\alpha_j + \varepsilon) \cdot d_j + \varepsilon \cdot d_{\text{bad}}\Big] \\
&= d_i(1 - \frac{1}{n}) + \frac{1}{n} \cdot \beta\Big[\sum_j \big(P_{ij}\alpha_j d_j + \varepsilon d_{\text{bad}}(1 + \lambda(U))\Big] \\
&= d_i(1 - \frac{1}{n}) + \frac{1}{n} \cdot \beta\Big[\lambda_1(U) d_i + \varepsilon d_{\text{bad}}(1 + \lambda(U))\Big] \\
&\le d_i \exp\Big(-\frac{1}{n}\Big(1 - \beta\big(\lambda_1(U) + \varepsilon\frac{d_{\text{bad}}}{d_i}(1 + \lambda_1(U))\big)\Big)\Big).
\end{aligned} \tag{22}$$

By the choice of ε we then have

$$\beta\big(\lambda_1(U) + \varepsilon\frac{d_{\text{bad}}}{d_i}(1 + \lambda_1(U)\big) \le \beta\big(\lambda_1(U) + \frac{\varepsilon_0}{4}\big) \le (\lambda_1(W) + \frac{\varepsilon_0}{2})\beta < 1. \tag{23}$$

Hence using (23) in (22) we have

$$\mathbb{E}[\ell(X_1, Y_1)] \le d_i \exp(-\frac{1}{n}c),$$

where $c = 1 - \beta\big(\lambda_1(U) + \varepsilon\frac{d_{\text{bad}}}{d_i}(1 + \lambda_1(U))\big) > 0$ and so from Lemma 16 we have the theorem.

Case II: v is BAD: In this case we will consider that v is BAD w.r.t. the labeling and so it can be connected to all the vertices in the worst case. Using similar discussion for case I we have:

- If $w = v$, then $d(X_1, Y_1) = 0$.
- If $w \notin \mathcal{N}(v) \cup \{v\}$, then $d(X_1, Y_1) = d_{\text{bad}}$.
- If $w \in \mathcal{N}(v)$ and w gets label j by the labeling, then w.p. $p(\tau, v) - p(\sigma, v)$,
  - $\ell(X_1, Y_1) = d_{\text{bad}} + d_j$, if w is GOOD.
  - $\ell(X_1, Y_1) = d_{\text{bad}} + d_{\text{bad}}$, if w is BAD.

Similarly we have,

$$\begin{aligned}
\mathbb{E}[\ell(X_1, Y_1)] &\le d_{\text{bad}}(1 - \frac{1}{n}) + \frac{1}{n} \cdot \tanh(\beta/n)\Big[\sum_j n d_j + \varepsilon n \cdot d_{\text{bad}}\Big] \\
&\le d_{\text{bad}}(1 - \frac{1}{n}) + \frac{1}{n} \cdot \beta\Big[\sum_j d_j + \varepsilon \cdot d_{\text{bad}}\Big] \\
&\le d_{\text{bad}}(1 - \frac{1}{n}) + \frac{1}{n} \cdot \beta \cdot d_{\text{bad}}\Big[\lambda_1(U) + \varepsilon\Big] \\
&\le d_{\text{bad}} \exp\Big(-\frac{1}{n}\Big(1 - \beta\big(\lambda_1(U) + \varepsilon\big)\Big)\Big).
\end{aligned}$$

By the choice of ε we then have

$$\beta(\lambda_1(U)+\varepsilon) \le \beta(\lambda_1(W)+\frac{\varepsilon_0}{2}) < 1.$$

Hence we will have the theorem from Lemma 16. ◀

8 Lower Bound for Mixing Time

Here we will prove the result about slow mixing of Theorem 5 using the well known conductance bound technique [7].

▶ **Lemma 18.** *[7] Let $\mathcal{M}$ be a Markov chain with state space Ω, transition matrix P, and stationary distribution μ. Let $A \subset \Omega$ such that $\mu(A) \le \frac{1}{2}$, and $B \subset \Omega$ that forms a barrier in the sense $P_{ij} = 0$ for $i \in A \setminus B$ and $j \in A^{\mathsf{c}} \setminus B$. Then the mixing time of $\mathcal{M}$ is at least $\frac{\mu(A)}{8\mu(B)}$.*

To find such sets we look at the sets with given signature or phase. Formally we define

$$A_\alpha := \{\sigma \big| |\{v \in V(G) | \sigma(v) = +\}| = \alpha n\}. \tag{24}$$

Now let Z_α denotes the partition function with signature α. To apply the Lemma 18 we consider $A = A_{<\frac{1}{2}} = \bigcup_{\alpha<\frac{1}{2}} A_\alpha$ and $B = A_{\frac{1}{2}}$. Trivially B is barrier between A and $A^{\complement}$. Now to show lower bound of $\frac{\mu(A)}{8\mu(B)}$ we give a lower bound on $\mu(A)$ and an upper bound on $\mu(B)$.

Lower bound on $\mu(A)$. Assume $\{G_n\}$ be a convergence sequence of dense graphs which converges to a graphon W, then the graphs also converge w.r.t. the microcanonical free energy, where microcanonical energy $\mathcal{F}_{\mathbf{a}}(W,\beta)$ is defined as

$$\mathcal{F}_{\mathbf{a}}(W,\beta) := \inf_{\rho:\alpha(\rho)=\mathbf{a}} \mathcal{E}(W,\beta,2\rho-1).$$

Now let's look at the free energy from (3):

$$\mathcal{F}(W,\beta) = \inf_{\rho:[0,1]\to[0,1]} \mathcal{E}(W,\beta,2\rho-1) = \mathcal{E}(W,\beta,2\rho^{opt}-1).$$

Now let's say we have $\int_{[0,1]} \rho^{opt}(x)\,dx = \alpha_c$ for some constant α_c (w.l.o.g., we can assume $\alpha_c < 1/2$). We denote $Z'_\alpha = Z(\beta)|_{\Omega_\alpha}$ and $Z_\alpha = Z(\beta)|_{A_\alpha}$, where Ω_α is defined in Section 2.3. Then from Proposition 4 we have :

$$\begin{aligned}
&\Big|\frac{1}{n}\log(Z'_{\alpha_c}) - \sup_{\int_{[0,1]}\rho(x)\,dx=\alpha_c} \big(-\mathcal{F}_{\alpha_c}(W,\beta)\big)\Big| < \varepsilon \\
&\Rightarrow \Big|\frac{1}{n}\log(Z'_{\alpha_c}) + \mathcal{E}(W,\beta,2\rho^{opt}-1)\Big| < \varepsilon \\
&\Rightarrow \frac{1}{n}\log(Z'_{\alpha_c}) > \mathcal{E}(W,\beta,2\rho^{opt}-1) - \varepsilon \\
&\Rightarrow \frac{1}{n}\log(Z_{<\frac{1}{2}}) > \frac{1}{n}\log(Z'_{\alpha_c}) > -\mathcal{E}(W,\beta,2\rho^{opt}-1) - \varepsilon \\
&\Rightarrow Z_{<\frac{1}{2}} > \exp(n(-\mathcal{E}(W,\beta,2\rho^{opt}-1)-\varepsilon)).
\end{aligned} \tag{25}$$

where we denote $Z_{<\frac{1}{2}} = \cup_{\alpha<\frac{1}{2}} Z_\alpha$.

Upper bound on $\mu(B)$: Suppose $\sup\limits_{\int_{[0,1]}\rho(x)=\frac{1}{2}} \mathcal{E}_{\frac{1}{2}}(W,\beta,2\rho-1) = \mathcal{E}(W,\beta,2\rho^*-1)$, for some ρ^*. Now from Proposition 4 we have that,

$$\begin{aligned}
&\left|\frac{1}{n}\log(Z'_{\frac{1}{2}}) - \sup_{\int_{[0,1]}\rho(x)=\frac{1}{2}} \big(-\mathcal{F}_{\frac{1}{2}}(W,\beta)\big)\right| < \varepsilon \\
\Rightarrow &\left|\frac{1}{n}\log(Z'_{\frac{1}{2}}) + \mathcal{E}(W,\beta,2\rho^*-1)\right| < \varepsilon \\
\Rightarrow &\ \frac{1}{n}\log(Z'_{\frac{1}{2}}) < -\mathcal{E}(W,\beta,2\rho^*-1) + \varepsilon \\
\Rightarrow &\ \frac{1}{n}\log(Z_{\frac{1}{2}}) < \frac{1}{n}\log(Z'_{\frac{1}{2}}) < -\mathcal{E}(W,\beta,2\rho^*-1) + \varepsilon \\
\Rightarrow &\ Z_{\frac{1}{2}} < \exp(n(-\mathcal{E}(W,\beta,2\rho^*-1) + \varepsilon)). \qquad (26)
\end{aligned}$$

Proof of Theorem 5.2. From (25) and (26) we have that

$$\begin{aligned}
\frac{\mu(A)}{8\mu(B)} &\geq \frac{\exp(n(-\mathcal{E}(W,\beta,2\rho^{opt}-1)-\varepsilon))}{\exp(n(-\mathcal{E}(W,\beta,2\rho^*-1)+\varepsilon))} \\
&= \exp(n(-\mathcal{E}(W,\beta,2\rho^{opt}-1) + \mathcal{E}(W,\beta,2\rho^*-1) - 2\varepsilon)) \\
&= \exp(n(c-2\varepsilon)). \qquad (27)
\end{aligned}$$

where $c = -\mathcal{E}(W,\beta,2\rho^{opt}-1) + \mathcal{E}(W,\beta,2\rho^*-1)$. As in this case we have $\beta\lambda_1(W) > 1$ and so from Theorem 6.2 we have $c > 0$. Hence choosing ε sufficiently small we obtain the theorem. ◀

9 Counterexample at Critical Temperature

9.1 Example of Fast Mixing at Critical Temperature

In this section we show a sequence of graphs $\{G_n\}$ such that $\{G_n\} \to W$ and we assume $\lambda_1(W)\beta = 1$. But the mixing time of Glauber dynamics on G_n is $O(n\log n)$. To show this we consider the graphs sampled from the model $\mathcal{G}(n, \frac{1}{2} - \frac{1}{\log n})$. Note that, if G_n is sampled from the model $\mathcal{G}(n, \frac{1}{2} - \frac{1}{\log n})$ then $\{G_n\} \to W$, where W is constant function such that $W(x,y) = \frac{1}{2}$ for all x, y. So we assume $\beta = 2$. By Chernoff bound it can be shown that w.h.p. for each vertex we have the number of neighbors of v is $\leq \frac{n}{2}$. Hence following the same path coupling defined in Section 7, we get fast mixing.

9.2 Example of Slow Mixing at Critical Temperature

In this section we show a sequence of graphs $\{G_n\}$ such that $\{G_n\} \to W$ and we assume $\lambda_1(W)\beta = 1$. But the mixing time of Glauber dynamics on G_n is super-polynomial (more precisely, $\exp(\Omega(\sqrt{n}))$). To show this we consider the graphs sampled from the model $\mathcal{G}(n, \frac{1}{2} + \frac{1}{\log n})$. Note that, if G_n is sampled from the model $\mathcal{G}(n, \frac{1}{2} + \frac{1}{\log n})$ then $\{G_n\} \to W$, where W is constant function such that $W(x,y) = \frac{1}{2}$ for all x, y. So we assume $\beta = 2$.

9.2.1 Properties of Random Graph

▶ **Lemma 19.** *Given a graph $G(= (V,E)) \sim \mathcal{G}(n, \frac{1}{2} + \frac{1}{\log n})$. Assume $S \subset V$ such that $|S| = \frac{n}{2} + k$, for some $k \geq 0$. Then we have w.h.p.:*

1. $E(S,S^{\mathsf{c}}) = (\frac{1}{2}+\frac{1}{\log n})(\frac{n^2}{4}-k^2)[1\pm\frac{c}{\sqrt{n}}]$.
2. $E(S,S) = (\frac{1}{2}+\frac{1}{\log n})(\binom{n/2+k}{2})[1\pm\frac{c}{\sqrt{n}}]$.
3. $E(S^{\mathsf{c}},S^{\mathsf{c}}) = (\frac{1}{2}+\frac{1}{\log n})(\binom{n/2-k}{2})[1\pm\frac{c}{\sqrt{n}}]$.

Proof. The lemma follows from Chernoff bound. ◀

Upper Bound on Balanced Configuration: From Lemma 19 we have w.h.p. for balanced configurations we have

$$\begin{aligned}
\mu(B) &\le \binom{n}{n/2}\exp(2\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})\binom{n/2}{2}[1+\frac{c}{\sqrt{n}}])\exp(-\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})\frac{n^2}{4}[1-\frac{c}{\sqrt{n}}])\\
&= \binom{n}{n/2}\exp(\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[2\binom{n/2}{2}(1+\frac{c}{\sqrt{n}})-\frac{n^2}{4}(1-\frac{c}{\sqrt{n}})])\\
&\le 2^n\exp(\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[\frac{n^2}{4}(1+\frac{c}{\sqrt{n}}-1+\frac{c}{\sqrt{n}})])\\
&= 2^n\exp(c\sqrt{n}[\frac{1}{2}+\frac{1}{\log n}]). \qquad (28)
\end{aligned}$$

Lower Bound on Unbalanced Configuration: Similarly from Lemma 19 we have w.h.p. for configurations with $(\frac{n}{2}+k)$ pluses and $(\frac{n}{2}-k)$ minuses we have

$$\begin{aligned}
\mu(A) &\ge \binom{n}{n/2+k}\exp(\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[\binom{n/2+k}{2}+\binom{n/2-k}{2}]\cdot[1-\frac{c}{\sqrt{n}}])\\
&\cdot\exp(-\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[\frac{n^2}{4}-k^2][1+\frac{c}{\sqrt{n}}])\\
&= \binom{n}{n/2+k}\exp(\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[\frac{n^2}{4}+k^2]\cdot[1-\frac{c}{\sqrt{n}}])\\
&\cdot\exp(-\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[\frac{n^2}{4}-k^2][1+\frac{c}{\sqrt{n}}])\\
&= \binom{n}{n/2+k}\exp(\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[(\frac{n^2}{4}+k^2)\cdot(1-\frac{c}{\sqrt{n}})-(\frac{n^2}{4}-k^2)(1+\frac{c}{\sqrt{n}})])\\
&= \binom{n}{n/2+k}\exp(\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[2(-\frac{n^{3/2}c}{4}+k^2)])\\
&\ge \frac{1}{\sqrt{\pi n/2}}2^n\exp(-\frac{2k^2}{n}-\frac{4k^3}{n^2})\exp(\frac{2}{n}(\frac{1}{2}+\frac{1}{\log n})[2(-\frac{n^{3/2}c}{4}+k^2)]). \qquad (29)
\end{aligned}$$

Here we use the fact that,

$$\binom{n}{n/2+k}\ge\frac{1}{\sqrt{\pi n/2}}2^n\exp(-\frac{2k^2}{n}-\frac{4k^3}{n^2}).$$

Now taking $k=c_1n^{7/8}$ we have from (28) and (29) we have:

$$\begin{aligned}
\frac{\mu(A)}{\mu(B)} &\ge \frac{1}{\sqrt{\pi n/2}}\exp(-\frac{2k^2}{n}-\frac{4k^3}{n^2})\cdot\exp([\frac{1}{2}+\frac{1}{\log n}](-2c\sqrt{n}+\frac{4k^2}{n}))\\
&= \frac{1}{\sqrt{\pi n/2}}\exp(-\frac{2k^2}{n}-\frac{4k^3}{n^2}-c\sqrt{n}-\frac{2c\sqrt{n}}{\log n}+\frac{2k^2}{n}+\frac{4k^2}{n\log n})
\end{aligned}$$

$$= \frac{1}{\sqrt{\pi n/2}} \exp(\frac{4k^2}{n}[\frac{1}{\log n} - \frac{c_1}{n^{1/8}}] - c\sqrt{n} - \frac{2c\sqrt{n}}{\log n})$$
$$= \frac{1}{\sqrt{\pi n/2}} \exp(\Omega(\sqrt{n})).$$

Hence we have the lower bound.

Acknowledgements. The authors thank the anonymous referees for the helpful suggestions.

References

1 Noam Berger, Claire Kenyon, Elchanan Mossel, and Yuval Peres. Glauber dynamics on trees and hyperbolic graphs. *Probability Theory and Related Fields*, 131(3):311–340, 2005.

2 Antonio Blanca and Alistair Sinclair. Dynamics for the Mean-field Random-cluster Model. In Naveen Garg, Klaus Jansen, Anup Rao, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 528–543, Dagstuhl, Germany, 2015. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.528.

3 Christian Borgs, Jennifer T. Chayes, László Lovász, Vera T. Sós, and Katalin Vesztergombi. Convergent sequences of dense graphs I: Subgraph frequencies, metric properties and testing. *Advances in Mathematics*, 219(6):1801–1851, 2008.

4 Christian Borgs, Jennifer T. Chayes, László Lovász, Vera T. Sós, and Katalin Vesztergombi. Convergent sequences of dense graphs II. Multiway cuts and statistical physics. *Annals of Mathematics*, 176(1):151–219, 2012.

5 Russ Bubley and Martin E. Dyer. Path coupling: A technique for proving rapid mixing in Markov chains. In *38th Annual Symposium on Foundations of Computer Science, FOCS'97, Miami Beach, Florida, USA, October 19-22, 1997*, pages 223–231, 1997. doi:10.1109/SFCS.1997.646111.

6 Paul Cuff, Jian Ding, Oren Louidor, Eyal Lubetzky, Yuval Peres, and Allan Sly. Glauber dynamics for the mean-field Potts model. *Journal of Statistical Physics*, 149(3):432–477, 2012.

7 Martin Dyer, Alan Frieze, and Mark Jerrum. On counting independent sets in sparse graphs. *SIAM Journal on Computing*, 31(5):1527–1541, 2002.

8 Andreas Galanis, Daniel Štefankovic, and Eric Vigoda. Swendsen-Wang Algorithm on the Mean-Field Potts Model. In Naveen Garg, Klaus Jansen, Anup Rao, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 815–828, Dagstuhl, Germany, 2015. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.APPROX-RANDOM.2015.815.

9 Andreas Galanis, Daniel Štefankovič, and Eric Vigoda. Inapproximability of the partition function for the antiferromagnetic Ising and hard-core models. *Combinatorics, Probability and Computing*, 25(04):500–559, 2016.

10 Andreas Galanis, Daniel Štefankovič, Eric Vigoda, and Linji Yang. Ferromagnetic Potts model: Refined #BIS-hardness and related results. *SIAM Journal on Computing*, 45(6):2004–2065, 2016.

11 Roy J. Glauber. Time-dependent statistics of the Ising model. *Journal of mathematical physics*, 4(2):294–307, 1963.

12 Ernst Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, 1925.

13 Mark Jerrum and Alistair Sinclair. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal on computing*, 22(5):1087–1116, 1993.

14 Wilhelm Lenz. Beitrag zum Verständnis der magnetischen Erscheinungen in festen Körpern. *Z. Phys.*, 21:613–615, 1920.

15 David A. Levin, Malwina J. Luczak, and Yuval Peres. Glauber dynamics for the mean-field Ising model: cut-off, critical power law, and metastability. *Probability Theory and Related Fields*, 146(1):223–265, 2010.

16 David Asher Levin, Yuval Peres, and Elizabeth Lee Wilmer. *Markov chains and mixing times*. American Mathematical Soc., 2009.

17 László Lovász. *Large networks and graph limits*, volume 60. American Mathematical Soc., 2012.

18 László Lovász and Vera T. Sós. Generalized quasirandom graphs. *Journal of Combinatorial Theory, Series B*, 98(1):146–163, 2008.

19 László Lovász and Balázs Szegedy. Limits of dense graph sequences. *Journal of Combinatorial Theory, Series B*, 96(6):933–957, 2006.

20 Fabio Martinelli. Lectures on Glauber dynamics for discrete spin models. *Lectures on probability theory and statistics*, pages 93–191, 2004.

21 Fabio Martinelli, Alistair Sinclair, and Dror Weitz. Glauber dynamics on trees: boundary conditions and mixing time. *Communications in Mathematical Physics*, 250(2):301–334, 2004.

22 Elchanan Mossel and Allan Sly. Exact thresholds for Ising–Gibbs samplers on general graphs. *The Annals of Probability*, 41(1):294–328, 2013.

23 Elchanan Mossel, Dror Weitz, and Nicholas Wormald. On the hardness of sampling independent sets beyond the tree threshold. *Probability Theory and Related Fields*, 143(3-4):401–439, 2009.

24 Kevin P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

25 Wolfgang Paul and Jörg Baschnagel. *Stochastic processes*. Springer, 1999.

26 Alistair Sinclair, Piyush Srivastava, Daniel Štefankovič, and Yitong Yin. Spatial mixing and the connective constant: Optimal bounds. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1549–1563. Society for Industrial and Applied Mathematics, 2015.

27 Alistair Sinclair, Piyush Srivastava, and Yitong Yin. Spatial mixing and approximation algorithms for graphs with bounded connective constant. In *54th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 300–309, 2013.

28 Allan Sly and Nike Sun. The computational hardness of counting in two-spin models on d-regular graphs. In *53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 361–369. IEEE, 2012.

10 Appendix

10.1 Proof of Theorem 6

Here we will prove the remaining proofs of Theorem 6. Before moving on to the proofs we need the following standard definitions.

10.1.1 Preliminaries

We will use the following with X being the space of measurable functions $[0,1] \to [0,1]$ and X^* will be the dual space.

▶ **Definition 20** (Weak Convergence). Let X be a normed space. Then a sequence $\{f_n\}$ in X is said to be weak-convergent to $f \in X$ if

$$\forall L \in X^* \text{ we have } L(f_n) \to L(f) \text{ as } n \to \infty,$$

and we denote this by $f_n \underset{w}{\to} f$.

▶ **Definition 21** (Weak Compactness). Let $(X, \|\cdot\|)$ be a normed space with dual space X^*. Then a set $M \subset X$ is called weak compact, if every sequence in M has a weak convergent subsequence with limit in M.

Next we will state two facts and a lemma which we will use in the proof of main theorem.

▶ **Fact 22.** *Let $\{f_n\}$ be a sequence in X such that $f_n \underset{w}{\to} f$ then $\langle f_n, h\rangle \to \langle f, h\rangle$ for all $h \in X$.*

▶ **Fact 23.** *The set of measurable function from $[0,1]$ to $[0,1]$ are weak-compact.*

▶ **Lemma 24.** *Let $f_n \underset{w}{\to} f$ then we have*

$$\limsup_{n\to\infty} H(f_n) \le H(f).$$

▶ **Definition 25** (Smoothed Function). Let U be a step graphon with steps $S_1, \ldots, S_k$ and f be a measurable function such that $f : [0,1] \to [0,1]$. Then smoothed version of f w.r.t. U is defined by the step function g with the step $S_1, \ldots, S_k$ as follows:

$$g(x) = c_i \text{ if } x \in S_i,$$

where $c_i = \sqrt{\frac{\langle f_n, T_{W_n} f_n\rangle}{\int_{S_i \times S_i} W(x,y)\,dx\,dy}}$.

▶ **Fact 26.** *Let f be a measurable function from $[0,1]$ to $[0,1]$ and g_n be the smoothed version of f w.r.t. the step graphon W_n. Then*

1. $\langle f, T_{W_n} f\rangle = \langle g_n, T_{W_n} g_n\rangle$,
2. $\mathrm{Ent}(f_n) \le \mathrm{Ent}(g_n)$.

10.1.2 Proof of Lemma 11

Proof of Lemma 11. Let's denote the value of the objective function at infimum by D, i.e.,

$$D = \inf_{m:[0,1]\to[-1,1]} \mathcal{E}(W, \beta, m) = \inf_{m:[0,1]\to[-1,1]} \Big(-\frac{\beta}{2}\langle m, T_W m\rangle - \mathrm{Ent}(m)\Big),$$

where

$$\begin{aligned}\mathrm{Ent}(m) = &-\int_0^1 \frac{1}{2}(1-m(x))\log(\frac{1}{2}(1-m(x)))\,dx\\ &-\int_0^1 \frac{1}{2}(1+m(x))\log(\frac{1}{2}(1+m(x)))\,dx.\end{aligned}$$

Let $\{m'_n\}$ be the sequence of functions such that $-\frac{\beta}{2}\langle m'_n, T_W m'_n\rangle - \mathrm{Ent}(m'_n) \to D$. By weak compactness of the set of measurable functions we know that there exists a subsequence $\{m_n\}$ of $\{m'_n\}$ and a measurable function m such that $-\frac{\beta}{2}\langle m_n, T_W m_n\rangle \to -\frac{\beta}{2}\langle m, T_W m\rangle$. From [18] we also have for any graphon W there exist a sequence of step graphons $\{W_t\}$'s such that $W_t \to W$ in L_1 distance. Hence we can also write $-\frac{\beta}{2}\langle m, T_{W_t} m\rangle \to -\frac{\beta}{2}\langle m, T_W m\rangle$. Let g_t be the smoothed version of m w.r.t. the step graphon W_t as defined in Definition 25. Using Fact 26 and the weak compactness of the set of measurable functions there exists a function g such that

$$\langle m, T_{W_t} m\rangle = \langle g_t, T_{W_t} g_t\rangle \to \langle g, T_W g\rangle. \tag{30}$$

Now let's assume another function $\rho : [0,1] \to [0,1]$ such that $\rho(x) = \frac{1}{2}(1 - m(x))\ \forall x \in [0,1]$. Also we define the functional for any measurable $\rho : [0,1] \to [0,1]$ as $H(\rho) = -\int_{[0,1]} \rho(x)\log(\rho(x))\,dx$. Hence $\mathrm{Ent}(m) = H(\rho) + H(1-\rho)$. Now Now from Lemma 24 and Fact 26 we have

$$\mathrm{Ent}(g) \geq \limsup_{n\to\infty} \mathrm{Ent}(g_t) \geq \limsup_{t\to\infty} \mathrm{Ent}(m). \tag{31}$$

Hence from (30) and (31) we have

$$\begin{aligned}\liminf_{t\to\infty}\Big(-\frac{\beta}{2}\langle m, T_{W_t} m\rangle - \mathrm{Ent}(m)\Big) &\geq -\frac{\beta}{2}\langle g, T_W g\rangle - \mathrm{Ent}(g)\\ D &\geq -\frac{c}{2}\langle g, T_W g\rangle - \mathrm{Ent}(g). \qquad (32)\end{aligned}$$

Hence the optimum is achieved for g which is a measurable function from $[0,1]$ to $[0,1]$ by weak*-compactness of the set. Hence the infimum is achieved. ◀

10.1.3 Remaining Proofs for Theorem 6.1

We need to prove the strict convexity of the functional defined in Lemma 12. Taking $\rho(x) := \frac{1}{2}(m(x)+1)$ in (3) we have

$$\begin{aligned}\mathcal{E}(W,\beta,m) &= -\frac{\beta}{2}\langle m, T_W m\rangle - \mathrm{Ent}(m)\\ &= -\frac{\beta}{2}\langle(2\rho-1), T_W(2\rho-1)\rangle - \mathrm{Ent}(2\rho-1)\\ &= \underbrace{-\frac{\beta}{2}\int_{[0,1]^2}(2\rho(x)-1)(2\rho(y)-1)W(x,y)\,dx\,dy}_{I(\rho)} - \mathrm{Ent}(2\rho-1). \qquad (33)\end{aligned}$$

We use this re-parameterization of the function as the function ρ is an eigenvector for the operator T_W and we will use the property of the eigenvector in the proof.

Proof of Lemma 14. We assume $\rho(x) := \frac{1}{2}(m(x)+1)$ and $s(x) := \frac{1}{2}(p(x)+1)$. Now from (33) for any $\alpha \in [0,1]$ we have the following lemma about the functional I defined in (33).

▶ **Lemma 27.** *For any $\rho, s : [0,1] \to [0,1]$ ($\rho \neq s$ up to measurability) and any $\alpha \in [0,1]$ we have*

$$I((1-\alpha)\rho + \alpha s) - (1-\alpha)I(\rho) - \alpha I(s) < 2\alpha(1-\alpha)||\rho - s||_2^2.$$

Also for the Ent functional we have the following lower bound.

▶ **Lemma 28.** *For any $\rho, s : [0,1] \to [0,1]$ and any $\alpha \in [0,1]$ we have*

$$\mathrm{Ent}((1-\alpha)(2\rho-1) + \alpha(2s-1)) - (1-\alpha)\mathrm{Ent}(2\rho-1) - \alpha\mathrm{Ent}(2s-1) \geq 2\alpha(1-\alpha)||\rho - s||_2^2.$$

From the statement of Lemma 27 and 28, Lemma 12 directly follows. ◀

Now we finish the remaining proofs.

Proof of Lemma 27. From (33) we have

$$\begin{aligned}&I((1-\alpha)\rho+\alpha s)-(1-\alpha)I(\rho)-\alpha I(s)\\&=-\frac{\beta}{2}\Big[\int_{[0,1]^2}(2((1-\alpha)\rho(x)+\alpha s(x))-1)(2((1-\alpha)\rho(y)+\alpha s(y))-1)W(x,y)\,dx\,dy\\&-\int_{[0,1]^2}\Big((1-\alpha)(2\rho(x)-1)(2\rho(y)-1)-\alpha(2s(x)-1)(2s(y)-1)\Big)W(x,y)\,dx\,dy\Big]\\&=\frac{\beta}{2}\int_{[0,1]^2}\Big[4\alpha(1-\alpha)[\rho(x)\rho(y)+s(x)s(y)-2\rho(x)s(y)]\Big]W(x,y)\,dx\,dy\\&=2\beta\alpha(1-\alpha)\int_{[0,1]^2}\Big[(\rho(x)-s(x))(\rho(y)-s(y))\Big]W(x,y)\,dx\,dy.\end{aligned}\tag{34}$$

Now in (34) we use the fact that $\lambda_1(W)$ is the largest eigenvalue of the graphon W as defined in Definition 3 and also $\lambda_1(W)\beta<1$. So we can rewrite (34) as,

$$\begin{aligned}&I((1-\alpha)\rho+\alpha s)-(1-\alpha)I(\rho)-\alpha I(s)\\&=2\alpha(1-\alpha)\beta\int_{[0,1]}(\rho(y)-s(y))\Big[\int_{[0,1]}(\rho(x)-s(x))W(x,y)\,dx\Big]\,dy\\&\le 2\alpha(1-\alpha)(\lambda_1(W)\beta)\int_{[0,1]}(\rho(y)-s(y))^2\,dy<2\alpha(1-\alpha)||\rho-s||_2^2.\end{aligned}\tag{35}$$

This completes the proof. ◀

Proof of Lemma 28. To prove the lemma we will use the following lemma as the main tool.

▶ **Lemma 29.** *Let $\alpha\in[0,1]$ and $R,S\in(0,1)$. Then we have*

$$\begin{aligned}-(1-\alpha)R\ln(1+\frac{\alpha}{R}(S-R))-\alpha S\ln(1+\frac{1-\alpha}{S}(R-S))\\-(1-\alpha)(1-R)\ln(1+\frac{\alpha}{1-R}(R-S))-\alpha(1-S)\ln(1+\frac{1-\alpha}{1-S}(S-R))\\\ge 2\alpha(1-\alpha)(S-R)^2.\end{aligned}$$

Now we apply Lemma 29 for each point of the integral, i.e., we set $R=\rho(x)$ and $S=s(x)$ and taking integral over $[0,1]$ we have

$$\begin{aligned}&\mathrm{Ent}((1-\alpha)(2\rho-1)+\alpha(2s-1))-(1-\alpha)\mathrm{Ent}(2\rho-1)-\alpha\mathrm{Ent}(2s-1)\\&=-(1-\alpha)\int_{[0,1]}\Big[\rho(x)\ln(1+\frac{\alpha}{\rho(x)}(s(x)-\rho(x)))-\alpha s(x)\ln(1+\frac{1-\alpha}{s(x)}(\rho(x)-s(x)))\\&-(1-\alpha)(1-\rho(x))\ln(1+\frac{\alpha}{1-\rho(x)}(\rho(x)-s(x)))\\&-\alpha(1-s(x))\ln(1+\frac{1-\alpha}{1-s(x)}(s(x)-\rho(x)))\Big]\,dx\\&\ge 2\alpha(1-\alpha)\int_{[0,1]}(\rho(x)-s(x))^2\,dx=2\alpha(1-\alpha)||\rho-s||_2^2.\end{aligned}$$

This completes the proof of Lemma 28. ◀

Now we state another lemma which is used to prove Lemma 29.

▶ **Lemma 30.** *Let $R \in (0,1)$ and $x \in (-R, 1-R)$. Then*

$$F(R,x) := -R\ln(1+\frac{x}{R}) - (1-R)\ln(1-\frac{x}{1-R}) - 2x^2 \geq 0. \tag{36}$$

Proof. We have $F(R,x) = F(1-R,-x)$ and hence it is enough to show (36) for $x \geq 0$. Note that

$$F(R,0) = 0 \quad \text{and} \quad \lim_{x\to(1-R)^-} F(R,x) = \infty. \tag{37}$$

We have that $F(R,x)$ is differentiable on $(-R, 1-R)$ with

$$\frac{\partial}{\partial x}F(R,x) = \frac{-x(2x+2R-1)^2}{(x+R)(x-(1-R))}.$$

If $R \geq 1/2$ there are no critical points of $F(R,x)$ on $(0, 1-R)$ and from (37) we get $F(R,x) \geq 0$ for $x \in (0,1-R)$. Now assume $R < 1/2$. The only critical point of $F(R,x)$ on $(0, 1-R)$ is $x = 1/2 - R$. We only need to prove that for all $R \in (0, 1/2)$

$$F(R, 1/2 - R) \geq 0.$$

It will be convenient to parameterize $R = 1/2 - T$. We have

$$F(1/2-T, T) = (1/2-T)\ln(1-2T) + (1/2+T)\ln(1+2T) - 2T^2 =: G(T).$$

Note that $G(0) = 0$ and

$$G'(T) = -\ln(1-2T) + \ln(1+2T) - 4T.$$

We will show $G'(T) \geq 0$ for $T \in [0, 1/2)$. Note that $G'(0) = 0$ and for $T \in [0, 1/2)$ we have

$$G''(T) = \frac{16T^2}{1-4T^2} \geq 0,$$

and hence $G'(T) \geq 0$ for $T \in [0, 1/2)$. ◀

Proof of Lemma 29. From Lemma 30 we have

$$(1-\alpha)F(R, \alpha(S-R)) + \alpha F(S, (1-\alpha(R-S)),$$

which is equivalent to the inequality we are proving. ◀

10.1.4 Remaining Proofs for Theorem 6.2

Recall that $S = \{\rho : [0,1] \to [0,1] | \int_{[0,1]} \rho(x)\,dx = \frac{1}{2}\}$. Also assume $A^l_\rho = \{x \in [0,1] | \rho(x) < \frac{1}{2}\}$ and $A^g_\rho = \{x \in [0,1] | \rho(x) > \frac{1}{2}\}$. Then note that A^l_ρ has positive measure if and only if A^g_ρ has positive measure. Also denote $A^{geq}_\rho = A^g_\rho \cup A^{eq}_\rho$, where $A^{eq}_\rho = \{x \in [0,1] | \rho(x) = \frac{1}{2}\}$.

▶ **Definition 31.** Given a function $\rho \in S$ we define another measurable function $\hat{\rho} : [0,1] \to [0,1]$ as follows:

$$\hat{\rho}(x) = \begin{cases} \rho(x) & \textit{if } x \in A^g_\rho, \\ 1-\rho(x) & \textit{otherwise.} \end{cases}$$

We have the following important property of $\hat{\rho}$:

▶ **Claim 32.** *If $x \in A^l_\rho$ then $\hat{\rho}(x) > \rho(x)$.*

▶ **Claim 33.** *Assume $\rho : [0,1] \to [0,1]$ is a measurable function and $\hat{\rho}$ as defined in definition 31. Then*

1. *$\hat{\rho}$ is also a measurable function.*
2. $\mathrm{Ent}(2\hat{\rho} - 1) = \mathrm{Ent}(2\rho - 1)$.

Proof of Claim 33.

1. Follows from the properties of measurability.
2. This follows from the symmetry of Ent function. ◀

Proof of Lemma 33. From Claim 33 we know that $\mathrm{Ent}(2\hat{\rho} - 1) = \mathrm{Ent}(2\rho - 1)$. Hence to prove (20) it is enough to prove that if $\rho \in S = \{\rho : [0,1] \to [0,1] | \int_{[0,1]} \rho(x)\,dx = \frac{1}{2}\} \setminus \{\rho : [0,1] \to [0,1] | f \underset{m}{\approx} \rho$ and $\rho(x) = \frac{1}{2} \forall x\}$, then

$$I(\hat{\rho}) < I(\rho).$$

where $I(\rho)$ is defined as in (33). This follows because $\rho(x) \leq \hat{\rho(x)}$ for all x and in particular $\rho(x) < \hat{\rho(x)}$, when $x \in A^l_\rho$ and also $W(x,y)$ is positive everywhere. ◀

Proof of Lemma 15. Let's consider the following function $\rho^b : [0,1] \to [0,1]$:

$$\rho^b(x) = \frac{1}{2}(1 + \varepsilon e_1(x)),$$

for all $x \in [0,1]$, where e_1 is the eigenfunction w.r.t. the largest eigenvalue of W, i.e., $\int_{[0,1]} W(x,y) e_1(y) dy = \lambda_1(W) e_1(x)$ and $\varepsilon > 0$ is some parameter. Now we have

$$\begin{aligned} I(\rho^b) = I(\frac{1}{2}(1 + \varepsilon e_(x)) &= -\frac{\beta}{2} \int_{[0,1]^2} (\varepsilon e_1(x))(\varepsilon e_1(y)) W(x,y)\,dx\,dy \\ &= -\varepsilon^2 \frac{\beta}{2} \int_{[0,1]^2} e_1(x) e_1(y) W(x,y)\,dx\,dy \\ &= -\varepsilon^2 \frac{\beta}{2} \lambda_1(W) \|e_1\|_2^2 < -\frac{\varepsilon^2}{2} \|e_1\|_2^2. \end{aligned} \tag{38}$$

Similarly for entropy we have:

$$\begin{aligned} \mathrm{Ent}(2\rho^b - 1) &= \mathrm{Ent}(\varepsilon \cdot e) \\ &= -\int_{[0,1]} \frac{1}{2}(1 + \varepsilon e_1(x)) \log(\frac{1}{2}(1 + \varepsilon e_1(x))) - \int_{[0,1]} \frac{1}{2}(1 - \varepsilon e_1(x)) \log(\frac{1}{2}(1 - \varepsilon e_1(x))) \\ &= -\log\frac{1}{2} - \int_{[0,1]} \frac{1}{2}(1 + \varepsilon e_1(x)) \log(1 + \varepsilon e_1(x)) - \int_{[0,1]} \frac{1}{2}(1 - \varepsilon e_1(x)) \log(1 - \varepsilon e_1(x)) \\ &= -\log\frac{1}{2} - \int_{[0,1]} \frac{1}{2}(1 + \varepsilon e_1(x))[\varepsilon e_1(x) - \varepsilon^2 e_1^2(x) + \varepsilon^3 e_1^3(x) - \cdots] \\ &\qquad - \int_{[0,1]} \frac{1}{2}(1 + \varepsilon e_1(x))[-\varepsilon e_1(x) - \varepsilon^2 e_1^2(x) - \varepsilon^3 e_1^3(x) - \cdots] \\ &\approx -\log\frac{1}{2} - \frac{\varepsilon^2}{2} \|e_1\|_2^2. \end{aligned} \tag{39}$$

Hence from (38) and (39) $\mathcal{E}(W, \beta, 2\rho^b - 1) \approx \mathcal{E}(W, \beta, 2\rho^{\frac{1}{2}} - 1) - c_\varepsilon \cdot \varepsilon^2$ which implies that $\mathcal{E}(W, \beta, 2\rho - 1)$ is decreasing in the given direction and we have the lemma. ◀

On the Expansion of Group-Based Lifts*

Naman Agarwal[1], Karthekeyan Chandrasekaran[2], Alexandra Kolla[3], and Vivek Madan[4]

1 **Princeton University, Princeton, Princeton, NJ, USA**
`namana@cs.princeton.edu`

2 **University of Illinois Urbana-Champaign, Urbana-Champaign, IL, USA**
`karthe@illinois.edu`

3 **University of Illinois Urbana-Champaign, Urbana-Champaign, IL, USA**
`akolla@illinois.edu`

4 **University of Illinois Urbana-Champaign, Urbana-Champaign, IL, USA**
`vmadan2@illinois.edu`

Abstract

A k-lift of an n-vertex base graph G is a graph H on $n \times k$ vertices, where each vertex v of G is replaced by k vertices $v_1, \ldots, v_k$ and each edge uv in G is replaced by a matching representing a bijection π_{uv} so that the edges of H are of the form $(u_i, v_{\pi_{uv}(i)})$. Lifts have been investigated as a means to efficiently construct expanders. In this work, we study lifts obtained from *groups and group actions*. We derive the spectrum of such lifts via the representation theory principles of the underlying group. Our main results are:

1. A uniform random lift by a cyclic group of order k of any n-vertex d-regular base graph G, with the nontrivial eigenvalues of the adjacency matrix of G bounded by λ in magnitude, has the new nontrivial eigenvalues bounded by $\lambda + \mathcal{O}(\sqrt{d})$ in magnitude with probability $1 - ke^{-\Omega(n/d^2)}$. The probability bounds as well as the dependency on λ are almost optimal. As a special case, we obtain that there is a constant c_1 such that for every $k \leq 2^{c_1 n/d^2}$, there exists a lift H of every Ramanujan graph by a cyclic group of order k such that H is *almost Ramanujan* (nontrivial eigenvalues of the adjacency matrix at most $O(\sqrt{d})$ in magnitude). We also show how this result leads to a quasi-polynomial time deterministic algorithm to construct almost Ramanujan expanders.
2. There is a constant c_2 such that for every $k \geq 2^{c_2 nd}$, there *does not* exist an *abelian k-lift* H of any n-vertex d-regular base graph such that H is almost Ramanujan. This can be viewed as an analogue of the well-known no-expansion result for constant degree abelian Cayley graphs.

Suppose k_0 is the order of the largest abelian group that produces expanding lifts. Our two results highlight lower and upper bounds on k_0 that are tight upto a factor of d^3 in the exponent, thus suggesting a threshold phenomenon.

1998 ACM Subject Classification G.2.2 Graph Theory

Keywords and phrases Expanders, Lifts, Spectral Graph Theory

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.24

1 Introduction

Expander graphs have spawned research in pure and applied mathematics during the last several years, with applications in multiple fields including complexity theory, robust computer networks, error-correcting codes, de-randomization, compressed sensing and metric

[*] A full version of the paper is available at `https://arxiv.org/abs/1311.3268`.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 24; pp. 24:1–24:13
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

embeddings [28, 16]. Informally, an expander is a graph in which every small subset of vertices has a relatively large edge boundary. Most applications are concerned with d-regular graphs. The largest eigenvalue of the adjacency matrix of d-regular graphs is d and is known as a trivial eigenvalue. In case of bipartite d-regular graphs, the largest and smallest eigenvalues of their adjacency matrix are d and $-d$ and these are referred to as trivial eigenvalues. The expansion of d-regular graphs is determined by the difference between d and the largest (in magnitude) non-trivial eigenvalue of the adjacency matrix, denoted λ. Roughly, the smaller λ is, the better the graph expansion. The Alon-Boppana bound ([25]) states that $\lambda \geq 2\sqrt{d-1} - o(1)$ for non-bipartite graphs. Thus, graphs with $\lambda \leq 2\sqrt{d-1}$ are optimal expanders and are called Ramanujan.

A simple probabilistic argument shows the existence of infinite families of expander graphs [26]. However, constructing such infinite families explicitly has proven to be a challenging and important task. It is easy to construct Ramanujan graphs with a small number of vertices: d-regular complete graphs and complete bipartite graphs are Ramanujan. The challenge is to construct an infinite family of d-regular graphs that are all Ramanujan, which was first achieved by Lubotzky, Phillips and Sarnak [19] and Margulis [23]. They built Ramanujan graphs from Cayley graphs. All of their graphs are regular, have degrees $p+1$ where p is a prime, and their proofs rely on deep number theoretic facts. In two breakthrough papers, Marcus, Spielman, and Srivastava showed the existence of bipartite Ramanujan graphs of all degrees [21, 22]. However they do not provide an efficient algorithm to construct those graphs. Cohen [7] adapted the techniques of [22]to design an efficient algorithm to construct Ramanujan multi-graphs. A striking result of Friedman [10] and a slightly weaker but more general result of Puder [27], shows that almost every d-regular graph on n vertices is very close to being Ramanujan, i.e., for every $\epsilon > 0$, asymptotically almost surely, $\lambda < 2\sqrt{d-1}+\epsilon$. It is still unknown whether the event that a random d-regular graph is exactly Ramanujan happens with constant probability. Despite a large body of work on the topic, all attempts to efficiently construct large Ramanujan expander (simple) graphs of any given degree have failed, and exhibiting such a construction remains an intriguing open problem.

A combinatorial approach to constructing expanders, initiated by Friedman [9], is to obtain new (larger) Ramanujan graphs from smaller ones. In this approach, we start with a base graph which is "lifted" to obtain a larger graph. Concretely, a k-lift of an n-vertex base-graph G is a graph H on $k \times n$ vertices , where each vertex u of G is replaced by k vertices $u_1, \ldots, u_k$ and each edge uv in G is replaced by a matching between $u_1, \ldots, u_k$ and $v_1, \ldots, v_k$. In other words, for each edge uv of G there is a permutation π_{uv} of k elements so that the corresponding k edges of H are of the form $u_i v_{\pi_{uv}(i)}$. The graph H is a (uniformly) *random* lift of G if for every edge uv the bijection π_{uv} is chosen uniformly at random from the set S_k of permutations of k elements.

Since we are focusing on Ramanujan graphs, we will restrict our attention to lifts of d-regular graphs. It is easy to see that any lift H of a d-regular base-graph G is itself d-regular and inherits all the eigenvalues of G. We will refer to the inherited eigenvalues as "old" eigenvalues and the rest of the eigenvalues as "new" eigenvalues. In order to use the lifts approach for constructing expanders, it is necessary that the lift also inherit the expansion properties of the base graph. Naturally, one hopes that a random lift of a Ramanujan graph will also be (almost) Ramanujan with high probability.

Friedman [9] first studied the eigenvalues of random k-lifts of regular graphs and proved that every new eigenvalue of H is $O(d^{3/4})$ with high probability. He conjectured a bound of $2\sqrt{d-1} + o(1)$, which would be tight (see, e.g. [14]). Linial and Puder [17] improved Friedman's bound to $O(d^{2/3})$. Lubetzky, Sudakov and Vu [18] showed that the magnitude of every nontrivial eigenvalue of the lift is $O(\lambda \log d)$, where λ is the largest (in magnitude) nontrivial eigenvalue of the base graph, thus improving on the previous results when G is significantly expanding. Adarrio-Berry and Griffiths [1] further improved the bounds above by showing that every new eigenvalue of H is $O(\sqrt{d})$, and Puder [27] proved the nearly-optimal bound of $2\sqrt{d-1}+1$. All those results hold with probability tending to 1 as $k \to \infty$, thus the order k of the lift in question needs to be large. Nearly no results were known in the regime where k is bounded with respect to the number of nodes n of the graph. A "relativized" version of the Alon-Boppana Conjecture regarding lower-bounding the new eigenvalues of lifts was also recently shown in [12] and [4].

Bilu and Linial [3] were the first to study k-lifts of graphs with bounded k, and suggested constructing Ramanujan graphs through a sequence of 2-lifts of a base graph: start with a small d-regular Ramanujan graph on some finite number of nodes (e.g. K_{d+1}). Every 2-lift operation doubles the number of vertices in the graph. If there is a way to preserve expansion after lifting, then repeating this operation will give large good expanders of the same bounded degree d. The authors in [3] showed that if the starting graph G is significantly expanding so that $\lambda(G) = O(\sqrt{d \log d})$, then there exists a random 2-lift of G that has all its new eigenvalues upper-bounded in magnitude by $O(\sqrt{d \log^3 d})$. In a recent breakthrough work, Marcus, Spielman and Srivastava [21] showed that for every bipartite d-regular graph G, there exists a 2-lift of G, such that the new eigenvalues achieve the Ramanujan bound of $2\sqrt{d-1}$. But their result still does not provide an efficient algorithm to find such lifts.

1.1 Our Results

In this work, we study the lifts approach to efficiently construct almost Ramanujan expanders of all degrees. We derive these lifts from groups. This is a natural generalization of Cayley graphs.

▶ **Definition 1** (Γ-lift). Let Γ be a group of order k with $\cdot$ denoting the group operation. A Γ-lift of an n-vertex base graph $G = (V, E)$ is a graph $H = (V \times \Gamma, E')$ obtained as follows: it has $k \times n$ vertices, where each vertex u of G is replaced by k vertices $\{u\} \times \Gamma$. For each edge uv of G, we choose an element $g_{uv} \in \Gamma$ and replace that edge by a perfect matching between $\{u\} \times \Gamma$ and $\{v\} \times \Gamma$ that is given by the edges $u_i v_j$ for which $g_{uv} \cdot i = j$.

We denote the order k of the group Γ to be the order of the lift. We refer to Γ-lifts obtained using $\Gamma = \mathbb{Z}/k\mathbb{Z}$, the additive group of integers modulo k, as shift k-lifts. Since every cyclic group of order k is isomorphic to $\mathbb{Z}/k\mathbb{Z}$, we have that Γ-lifts are shift k-lifts whenever Γ is a cyclic group of order k.

A tight connection between the spectrum of Γ-lifts and the representation theory of the underlying group Γ is known [24, 8]. This connection tells us that the lift incurs the eigenvalues of the base graph, while its new eigenvalues are the union of eigenvalues of a collection of matrices arising from the group elements assigned to the edges and the irreducible representations of the group. We note that this connection has also been recently used in [15] in the context of expansion of lifts, aiming to generalize the results in [22]. In this work, we address the expansion of Γ-lifts obtained from cyclic groups and abelian groups.

In order to understand the expansion properties of lifts, it suffices to focus on the new eigenvalues of the lifted graph by the above-mentioned connection. We present a high probability bound on the expansion of random shift k-lifts for bounded k.

▶ **Theorem 2.** *Let G be a d-regular n-vertex graph, where $2 \le d \le \sqrt{n/(3\ln n)}$, with largest (in magnitude) non-trivial eigenvalue λ, where $\lambda \ge \sqrt{d}$. Let H be a random shift k-lift of G*

with λ_{new} being the largest (in magnitude) new eigenvalue of H. Then

$$\lambda_{new} = O(\lambda)$$

with probability $1 - k \cdot e^{-\Omega(n/d^2)}$. Moreover, if G is moderately expanding such that $\lambda \leq d/\log d$, then

$$\lambda_{new} - \lambda = O(\sqrt{d})$$

with probability $1 - k \cdot e^{-\Omega(n/d^2)}$.

We say that a graph is *almost Ramanujan* if all its non-trivial eigenvalues are bounded by $O(\sqrt{d})$ in magnitude. By the above result, if the base graph G is Ramanujan, then the random shift k-lift will be almost Ramanujan with high probability.

Remark 1. In contrast to lifts of order k, where $k \to \infty$ when $n \to \infty$, the dependency of λ_{new} on λ is necessary for the case of bounded k. This has previously been observed by the authors in [3] who gave the following example: Let G be a disconnected graph on n vertices that consists of $n/(d+1)$ copies of K_{d+1}, and let H be a random 2-lift of G. Then the largest non-trivial eigenvalue of G is $\lambda = d$ and it can be shown that with high probability, $\lambda_{new} = \lambda = d$. Therefore, our eigenvalue bounds are nearly tight.

Specializing Theorem 2 for the case of 2-lifts gives the following Corollary which improves upon the multiplicative $\log d$ factor in the eigenvalue bound that is present in the result of Bilu-Linial [3].

► **Corollary 3.** *Let G be a d-regular n-vertex graph, where $2 \leq d \leq \sqrt{n/(3\ln n)}$, with largest (in magnitude) non-trivial eigenvalue λ, where $\lambda \geq \sqrt{d}$. Let H be a random 2-lift of G with λ_{new} being the largest (in magnitude) new eigenvalue of H. Then*

$$\lambda_{new} = O(\lambda)$$

with probability $1 - e^{-\Omega(n/d^2)}$. Moreover, if G is moderately expanding such that $\lambda \leq d/\log d$, then

$$\lambda_{new} - \lambda = O(\sqrt{d})$$

with probability $1 - e^{-\Omega(n/d^2)}$.

Remark 2. The multiplicative $\log d$ factor in the eigenvalue bound present in the result of Bilu-Linial [3] arises due to the use of the converse of the Expander Mixing Lemma along with an epsilon-net style argument in their analysis. The converse of the Expander Mixing Lemma is provably tight, so straightforward use of the converse will indeed incur the $\log d$ factor. We are able to improve the eigenvalue bound by performing a fine-grained analysis of the epsilon-net argument, avoiding direct use of the converse.

Lifts based on groups immediately suggest an algorithm towards building d-regular n-vertex Ramanujan expanders. In order to describe this algorithm, we first describe the brute-force algorithm that follows from the existential result of [21]. The approach is to start with the complete bipartite graph $K_{d,d}$ and lift the graph $\log_2(n/2d)$ times. At each stage, we do a brute-force search over the space of all possible 2-lifts and pick the best one (i.e., one with smallest new maximum eigenvalue in magnitude). However, since a graph (V, E) has $2^{|E|}$ possible 2-lifts, it follows that the final lift will be chosen from among $2^{nd/4}$ possible 2-lifts, which means that the brute force algorithm will run in time exponential in nd.

Next, suppose that for every $k \geq 2$, we are guaranteed the existence of a group Γ of order k such that for every base graph there exists a Γ-lift that has all its new eigenvalues at most $2\sqrt{d-1}$ in magnitude. For example, [5] suggests the possibility that for every k and for every base graph, there exists a shift k-lift that has all new eigenvalues with magnitude at most $2\sqrt{d-1}$. Then a brute force algorithm similar to the one above, would perform only one lift operation of the base graph $K_{d,d}$ to create a Γ-lift with $n = 2dk$ vertices. This algorithm would only have to choose the best among k^{d^2} possibilities (k different choices of group element per edge of the base graph), which is polynomial in n, the size of the constructed graph (here we have assumed that d is a constant). This motivates the following question: what is the largest possible group Γ that might produce expanding Γ-lifts? Our next result rules out the existence of large abelian groups that might lead to (even slightly) expanding lifts.

▶ **Theorem 4.** *For every n-vertex d-regular graph G, every real-value $\epsilon \in (0, 1/e)$, and every abelian group Γ of size at least*

$$k = exp\left(\frac{nd\log\frac{1}{\epsilon} + \log n}{\log\frac{1}{e\epsilon}}\right),$$

all Γ-lifts H of G has a new eigenvalue that is at least ϵd in magnitude. In particular, when $k = 2^{\Omega(nd)}$, there is no Γ-lift H of any n-vertex d-regular graph G all of whose eigenvalues are bounded by $O(\sqrt{d})$ in magnitude whenever Γ is an abelian group of order k.

Theorem 4 shows that we cannot expect to have arbitrarily large abelian groups with expanding lifts as suggested in [5].

Remark 3. The first and only known efficient construction of Ramanujan expander simple graphs are Cayley graphs of certain groups [19]. We observe that a Cayley graph for a group Γ with generator set S can be obtained as a Γ-lift of the bouquet graph (a graph that consists of one vertex with multiple self loops) [20]. Our no-expansion result for abelian groups complements the known result on no-expansion of abelian Cayley graphs [13].

Remark 4. Our Theorems 4 and 2 can be viewed as lower and upper bounds on the largest order k_0 of an abelian group Γ such that for every n-vertex graph, there exists a Γ-lift for which all new eigenvalues are $O(\sqrt{d})$. On the one hand, Theorem 2 shows that, for $k = 2^{O(n/d^2)}$, most of the shift k-lifts of a Ramanujan graph have their new eigenvalues to be $O(\sqrt{d})$. On the other hand, Theorem 4 shows that for $k = 2^{\Omega(nd)}$, there is no shift k-lift that achieves such eigenvalue guarantees. This suggests a threshold behavior for k_0.

We observe that Theorem 2 leads to a deterministic quasi-polynomial time algorithm for constructing almost Ramanujan families of graphs.

▶ **Theorem 5.** *There exists an algorithm that runs in time $2^{O(d^4 \log^2 n)}$ to construct a d-regular n-vertex graph such that all its non-trivial eigenvalues are $O(\sqrt{d})$ in magnitude.*

Proof. We use Algorithm 1. We note that the choice of r in the first step ensures that $r = O(d^2 \log n)$. By Theorem 2, there exists a lift G of the base graph G' such that

Algorithm 1 Quasi-polynomial time algorithm to construct expanders of arbitrary size n.

```
1: Pick an $r$ such that $r2^{cr/d^2} = n$, for a constant $c$ that appears in the eigenvalue bound in Theorem 2. Do an exhaustive search to find a $d$-regular graph $G'$ on $r$ vertices with $\lambda = O(\sqrt{d})$.
2: For $k = 2^{cr/d^2}$, do an exhaustive search to find a shift $k$-lift $G$ of the base graph $G'$ with minimum new eigenvalue (in magnitude).
```

$\lambda(G) = O(\sqrt{d})$. Thus, the exhaustive search in the second step gives a graph G whose non-trivial eigenvalues are $O(\sqrt{d})$ in magnitude.

In order to bound the running time, we note that the first step can be implemented to run in time $2^{O(r^2)} = 2^{O(d^4 \log^2 n)}$. To bound the running time of the second step, we observe that for each edge in G', there are k possible choices. Therefore, the size of the search space is at most $k^{rd/2} = 2^{cr^2/2d} = 2^{O(d^3 \log^2 n)}$ and for each k-lift, it takes $poly(n)$ time to compute $\lambda(G)$. Thus, the overall running time of the algorithm is $2^{O(d^4 \log^2 n)}$. ◀

Organization. We give some preliminary definitions, notations, facts and lemmas in Section 2. We prove Theorem 4 in Section 3. We illustrate the techniques behind proving Theorem 2 by presenting and proving a slightly weaker version of Theorem 2 (see Theorem 11) in Section 4. For proofs of the concentration inequality (Lemma 12) needed for the weaker version and Theorem 2, we refer the reader to the full version of the paper [2].

2 Preliminaries

In this section, we define certain notations and present the needed combinatorial inequalities and facts.

Notations. Let $G := (V, E)$ be a d-regular graph with n vertices. If G is d-regular bipartite, we will assume that the bipartition of the vertex set is given by $(\{1, \dots, n/2\}, \{n/2+1, \dots, n\})$. Let A be the adjacency matrix of G. Since A is a real symmetric matrix, its eigenvalues are also real. Let the eigenvalues of A be $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$. For a d-regular graph G, it is well-known that $\lambda_1 = d$. If G is bipartite, then $\lambda_n = -d$ and we define $\lambda_G := \max\{|\lambda_i| : i \in \{2, 3, \dots, n-1\}\}$. If G is non-bipartite, we define $\lambda_G := \max\{|\lambda_i| : i \in \{2, 3, \dots, n\}\}$. Thus, λ_G denotes the largest (in magnitude) non-trivial eigenvalue of G. When G is clear from the context, we will drop the subscript and simply write λ. For subsets $S, T \subseteq V$, let $E(S, T)$ be the number of edges $uv \in E$ with $u \in S$ and $v \in T$. We denote the largest eigenvalue of a matrix M by $\|M\|$ and the support of a vector x by $S(x)$. We define log() to be the log function with base 2. We represent e^x by $exp(x)$. Given a vector x whose coordinates are from $\{0, \pm 2^{-1}, \pm 2^{-2}, \dots, \pm 2^{-i}, \dots\}$ we define the *diadic decomposition* of x as the collection of vectors $\{2^{-i} u_i\}_{i \in \mathbb{Z}}$ where each u_i is a vector whose j'th coordinate is defined as

$$[u_i]_j := \begin{cases} 1 & \text{if } x_j = 2^{-i}, \\ -1 & \text{if } x_j = -2^{-i}, \\ 0 & \text{otherwise.} \end{cases}$$

▶ **Lemma 6** (Discretization Lemma). *Let $M \in \mathbb{R}^{n \times n}$ be a matrix with diagonal entries being* 0.

1. *For every $x \in \mathbb{R}^n$ with $||x||_\infty \leq 1/2$ there exists $y \in \{0, \pm 2^{-1}, \pm 2^{-2}, \ldots, \pm 2^{-i}, \ldots\}^n$ such that $|x^T Mx| \leq |y^T My|$ and $\|y\|^2 \leq 4\|x\|^2$. Moreover, each coordinate of x between 2^{-i} and $2^{-(i-1)}$ is rounded to either 2^{-i} or $2^{-(i-1)}$ and between -2^{-i} and $-2^{-(i-1)}$ is rounded to either -2^{-i} or $-2^{-(i-1)}$ in y.*
2. *For every $x_1, x_2 \in \mathbb{R}^n$ with $||x_1||_\infty, ||x_2||_\infty \leq 1/2$, there exist $y_1, y_2 \in \{0, \pm 2^{-1}, .., \pm 2^{-i}, ..\}^n$ such that $|x_1^T Mx_2| \leq |y_1^T My_2|$, $\|y_1\|^2 \leq 4\|x_1\|^2$, $\|y_2\|^2 \leq 4\|x_2\|^2$ and for $b \in \{1, 2\}$ each coordinate of x_b between 2^{-i} and $2^{-(i-1)}$ is rounded to either 2^{-i} or $2^{-(i-1)}$ and between -2^{-i} and $-2^{-(i-1)}$ is rounded to either -2^{-i} or $-2^{-(i-1)}$ in y_b.*

We need the following theorem showing that expanders have small diameter in order to show no-expansion of large abelian lifts.

▶ **Theorem 7** ([6]). *The diameter of a d-regular graph G with n vertices is at most $\frac{\log n}{\log(d/\lambda_G)}$.*

Lifts. We now state the relevant spectral properties of lifts (we derive the spectrum of general group-based lifts in the full version [2]). Some initial easy observations can be made about the structure of any lift: (i) the lifted graph is also regular with the same degree as the base graph and (ii) the eigenvalues of the adjacency matrix of the base graph are also eigenvalues of A_H. Therefore we call the n eigenvalues of the base graph as the *old* eigenvalues and the $n(k-1)$ other eigenvalues of A_H as the *new* eigenvalues. We will denote by λ_{new} the largest new eigenvalue of H in magnitude, which we also refer to as the "first" new eigenvalue for simplicity.

▶ **Definition 8** (Signing). Let $G = (V, E)$ be a base graph. Let E^f denote an arbitrary orientation of the edges of G and E^r denote the reverse orientation. Given a group Γ, a set S and an action $\cdot$ of Γ on S as in the Definition 1, we define a signing of G as a function $s : E^f \cup E^r \to \Gamma$ with the property that if $s(u, v) = g$ then $s(v, u) = g^{-1}$.

We observe that there is a bijection between signings and Γ-lifts. For the purposes of proving the results, we only need the spectrum of shift k-lifts. For a shift k-lift of a graph $G = (V, E)$ with adjacency matrix A, which is given by the signing $(s(i, j) = g_{i,j})_{(i,j) \in E}$, define the following family of Hermitian matrices $A_s(\omega)$ parameterized by ω where ω is a primitive k-th root of unity:

$$[A_s(\omega)]_{ij} = \begin{cases} 0 & \text{if } A_{ij} = 0, \textit{and} \\ \omega^{g_{i,j}} & \text{if } A_{ij} = 1. \end{cases}$$

The following lemma regarding the spectrum of shift k-lifts follows from classic results in representation theory.

▶ **Lemma 9.** *Let $G = (V, E)$ be a graph and H be a shift k-lift of G with the corresponding signing of the edges $(s(i, j) = g_{i,j})_{(i,j) \in E}$, where $g_{i,j} \in C_k$. Then the set of eigenvalues of H are given by*

$$\bigcup_{\omega:\ \omega \text{ is a primitive } k\text{-th root of unity}} \textit{eigenvalues}\,(A_s(\omega))\,.$$

The above simplifies significantly for 2-lifts as noted in the next corollary.

▶ **Corollary 10.** *When $k = 2$, the set of eigenvalues of a 2-lift H is given by the eigenvalues of A and the eigenvalues of A_s, where A_s is the signed adjacency matrix corresponding to the signing s, with entries from $\{0, 1, -1\}$.*

3 No-expansion of Abelian Lifts

In this section we show that it is impossible to find (even slightly) expanding graphs using lifts in large abelian groups Γ and thus prove Theorem 4 . By Theorem 7, we know that if a graph is an expander, then it has small diameter. We show that if the size of the (abelian) group Γ is large, then *all* Γ-lifts of any base graph have large diameter, and hence they cannot be expanders. We restate Theorem 4 for convenience.

▶ **Theorem 4.** *For every n-vertex d-regular graph G, every real-value $\epsilon \in (0, 1/e)$, and every abelian group Γ of size at least*

$$k = exp\left(\frac{nd\log\frac{1}{\epsilon} + \log n}{\log\frac{1}{e\epsilon}}\right),$$

all Γ-lifts H of G has a new eigenvalue that is at least ϵd in magnitude. In particular, when $k = 2^{\Omega(nd)}$, there is no Γ-lift H of any n-vertex d-regular graph G all of whose eigenvalues are bounded by $O(\sqrt{d})$ in magnitude whenever Γ is an abelian group of order k.

Proof. We prove the contrapositive. Let Γ be an abelian group of order k and $G = (V, E)$ be a base graph on n-vertices that is d-regular. Let $e_1, \dots, e_{nd/2}$ be an arbitrarily chosen ordering of the edges E. Let H be a lift graph obtained using a Γ-lift. Recall that the signing of the edges of the base graph correspond to group elements, which in turn correspond to permutations of k elements. Let these signing of the edges be $(\sigma_e)_{e\in E(G)}$. Let us define a layer L_i of H to be the set of vertices $\{v_i : v \in V\}$. We note that H has k layers.

Let us fix an arbitrary vertex v in G. Let Δ denote the diameter of H. Then, for every $j \in \{2, \dots, k\}$ there exists a path of length at most Δ in H from v_1 to a vertex in L_j. A layer j is reachable within distance Δ in H iff there exists a walk $e_1, e_2, \dots, e_t$ from v of length $t \le \Delta$ in G such that $\sigma_{e_t}\sigma_{e_{t-1}}\dots\sigma_{e_2}\sigma_{e_1}(1) = j$. Thus the set of layers reachable within distance Δ in H is contained in the set $S := \{\sigma_{e_t}\dots\sigma_{e_1}(1) : e_1, \dots, e_t \text{ is a walk from } v \text{ in } G \text{ of length } t \le \Delta\}$. Since the group Γ is abelian, $S \subseteq \{\sigma_{e_1}^{a_1}\sigma_{e_2}^{a_2}\dots\sigma_{e_{nd/2}}^{a_{nd/2}}(1) \mid \sum_{i=1}^{nd/2}|a_i| \le \Delta\} =: T$. Since H has k layers, the cardinality of S is at least k.

The number of integral a_i's satisfying $\sum_{i=1}^{nd/2}|a_i| \le \Delta$ is at most $\binom{(nd/2)+\Delta}{(nd/2)} \cdot 2^{(nd/2)}$. Therefore,

$$k \le |T| \le \binom{\frac{nd}{2}+\Delta}{\frac{nd}{2}}2^{\frac{nd}{2}} \le \left(2e\left(1+\frac{2\Delta}{nd}\right)\right)^{\frac{nd}{2}} \le (2e)^{\frac{nd}{2}}e^{\Delta}.$$

Since H has nk vertices, using Theorem 7, we have $\Delta \le (\log nk)/\log(d/\lambda(H))$. Thus, if $\lambda(H) \le \epsilon d$, then $\Delta \le (\log nk)/\log(1/\epsilon)$ and consequently,

$$k \le (2e)^{\frac{nd}{2}}e^{\frac{\log nk}{\log\frac{1}{\epsilon}}}.$$

Rearranging the terms, we obtain that

$$k \le (2e)^{\frac{nd}{2\left(1-\frac{1}{\log\frac{1}{\epsilon}}\right)}} exp\left(\frac{\log n}{\left(\log\frac{1}{\epsilon}\right)\left(1-\frac{1}{\log\frac{1}{\epsilon}}\right)}\right) \le exp\left(\frac{nd\log\frac{1}{\epsilon}+\log n}{\log\frac{1}{e\epsilon}}\right).$$ ◀

4 Expansion of Random 2-lifts: Overview

In this section, we illustrate the main techniques involved in proving Theorem 2 by stating and proving a slightly weaker version, namely Theorem 11. It focuses only on 2-lifts akin to Corollary 3 and is weaker in comparison to the eigenvalue bound in Corollary 3 by a multiplicative factor of four. The proof of this weaker result captures the main ideas involved in the proof of Theorem 2.

▶ **Theorem 11.** *Let G be a d-regular n-vertex graph, where $2 \leq d \leq \sqrt{n/(3\ln n)}$, with largest (in magnitude) non-trivial eigenvalue λ, where $\lambda \geq \sqrt{d}$. Let H be a random 2-lift of G with λ_{new} being the largest (in magnitude) new eigenvalue of H. Then,*

$$\lambda_{new} \leq 4\lambda + 10^{14} \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)$$

with probability at least $1 - e^{-n/d^2}$.

In order to prove this theorem, we use the concentration inequality in Lemma 12 (recall that for a vector x, its support is denoted by $S(x)$).

▶ **Lemma 12.** *Let G be a d-regular n-vertex graph, where $2 \leq d \leq \sqrt{n/(3\ln n)}$, with largest (in magnitude) non-trivial eigenvalue λ, where $\lambda \geq \sqrt{d}$. Let H be a random 2-lift of G with corresponding signed adjacency matrix A_s. The following statements hold with probability at least $1 - e^{-n/d^2}$:*

1. *For all $u_1, \ldots, u_r \in \{0, \pm1\}^n$, and $v_1, \ldots, v_\ell \in \{0, \pm1\}^n$ satisfying*

(I) *$S(u_i) \cap S(u_j) = \emptyset$ for every $i, j \in [r]$ and $S(v_i) \cap S(v_j) = \emptyset$ for every $i, j \in [\ell]$, and*

(II) *Either $|S(u_i)| > n/d^2$ for every $i \in [r]$ with non-zero u_i, or $|S(v_i)| > n/d^2$ for every $i \in [\ell]$ with non-zero v_i,*

we have

$$\left|\sum_{i \leq j} (2^{-i}u_i^T)A_s(2^{-j}v_j)\right| \leq 377\max(\sqrt{\lambda \log d}, \sqrt{d})\sum_{i=1}^{r}|S(u_i)|2^{-2i} + \left(\frac{\lambda}{5} + 10^{12}\sqrt{d}\right)\sum_{j=1}^{\ell}|S(v_j)|2^{-2j}.$$

3. *For all $u_1, \ldots, u_r \in \{0, \pm1\}^n$, and $v_1, \ldots, v_\ell \in \{0, \pm1\}^n$ satisfying* (I), (II) *and*

(III) *$|S(u_i)| > |S(v_j)|$ for every $i \in [r], j \in [\ell]$ with non-zero u_i,*

we have

$$\left|\sum_{i \leq j} (2^{-i}u_i^T)A_s(2^{-j}v_j)\right| \leq 31\max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\left(\sum_{i=1}^{r}|S(u_i)|2^{-2i} + \sum_{j=1}^{\ell}|S(v_j)|2^{-2i}\right).$$

We show the concentration inequality in Lemma 12 from Hoeffding's inequality by taking a suitable union bound (see the full version of the work [2] for a complete proof). We will now prove Theorem 11 using the lemma above. Our proof strategy resembles the proof strategy in [11].

Proof of Theorem 11. Let s denote the signing corresponding to H and A_s denote the signed adjacency matrix. By Corollary 10, the largest (in magnitude) new eigenvalue of the lift is $\lambda_{new} = \max_{x \in \mathbb{R}^n} |x^T A_s x|/x^T x$. To prove an upper bound on λ_{new}, we will bound $|x^T A_s x|/x^T x$ for all x with high probability. In particular, assuming that the events given by Lemma 12 hold, we will show that

$$|x^T A_s x| \leq 4\left(\lambda + 10^{13}\sqrt{d}\right)\|x\|^2.$$

By re-scaling we may assume that the maximum entry of x is less than 1/2 in absolute value. By Lemma 6, there exists a vector $y \in \{0, \pm 2^{-1}, \pm 2^{-2}, \ldots, \pm 2^{-i}, \ldots\}^n$ such that $|x^T A_s x| \leq |y^T A_s y|$ and $\|y\|^2 \leq 4\|x\|^2$. We will prove a bound on $|y^T A_s y|$ for every $y \in \{0, \pm 2^{-1}, \pm 2^{-2}, \ldots, \pm 2^{-i}, \ldots\}^n$, which in turn will imply the desired bound on $|x^T A_s x|$. Let us consider the diadic decomposition of $y = \sum_{i=1}^{\infty} 2^{-i} u_i$ obtained as follows: a coordinate of u_i is 1 if the corresponding coordinate of y is 2^{-i}, it is -1 if the corresponding coordinate of y is -2^{-i}, and is zero otherwise. We note that $S(u_i) \cap S(u_j) = \emptyset$ for every pair $i, j \in \mathbb{N}$.

Next, we partition the set of vectors u_i's based on their support sizes. Let $M := \{i \in \mathbb{N} : |S(u_i)| \leq n/d^2\}$ and $L := \{i \in \mathbb{N} : |S(u_i)| > n/d^2\}$ (we abbreviate M and L for mini and large supports respectively). Correspondingly, define $y_M := \sum_{i \in M} 2^{-i} u_i$ and $y_L = \sum_{i \in L} 2^{-i} u_i$. We note that $y = y_M + y_L$, $\|y\|^2 = \|y_M\|^2 + \|y_L\|^2 = \sum_{i \in \mathbb{N}} |S(u_i)| 2^{-2i}$, and

$$|y^T A_s y| \leq |y_M^T A_s y_M| + 2|y_M^T A_s y_L| + |y_L^T A_s y_L|.$$

We next bound each term in the following three claims.

▶ **Claim 13.**

$$|y_M^T A_s y_M| \leq \left(\lambda + \frac{8}{d}\right) \|y_M\|^2.$$

Proof. Let y'_M be a vector obtained from y_M by taking the absolute values of each entry. Then $\|y_M\|^2 = \|y'_M\|^2$ and $|y_M^T A_s y_M| \leq y_M'^T A y'_M$. Let $J = vv^T$ and $J' = v'v'^T$ where v is all ones vector and v' is defined as follows: $v'_i = 1$ for $1 \leq i \leq n/2$ and $v'_i = -1$ for $n/2 + 1 \leq i \leq n$. For non-bipartite graph G, we have

$$y_M'^T A y'_M = y_M'^T \left(A - \frac{d}{n} J\right) y'_M + y_M'^T \left(\frac{d}{n} J\right) y'_M \leq \lambda \|y'_M\|^2 + y_M'^T \left(\frac{d}{n} J\right) y'_M.$$

Above, we have used the fact that $A - \frac{d}{n}J$ has the same set of eigenvalues as A except for one – the eigenvalue d for the matrix A is translated to zero for the matrix $A - \frac{d}{n}J$. Similarly, for bipartite graphs, we have

$$\begin{aligned} y_M'^T A y'_M &= y_M'^T \left(A - \frac{d}{n} J + \frac{d}{n} J'\right) y'_M + y_M'^T \left(\frac{d}{n} J\right) y'_M - y_M'^T \left(\frac{d}{n} J'\right) y'_M \\ &\leq \lambda \|y'_M\|^2 + y_M'^T \left(\frac{d}{n} J\right) y'_M - y_M'^T \left(\frac{d}{n} J'\right) y'_M. \end{aligned}$$

Above, we have used the fact that $A - \frac{d}{n}J + \frac{d}{n}J'$ has the same set of eigenvalues as A except for two – the largest (in magnitude) two eigenvalues d for the matrix A are translated to zero for the matrix $A - \frac{d}{n}J + \frac{d}{n}J'$. It remains to bound $|y_M'^T \left(\frac{d}{n}J\right) y'_M|$ and $|y_M'^T \left(\frac{d}{n}J'\right) y'_M|$. Consider the diadic decomposition of $y'_M = \sum_{i \in M} 2^{-i} u'_i$, where the coordinates of u'_i are the absolute values of the coordinates of u_i.

$$\begin{aligned} \left| y_M'^T \left(\frac{d}{n} J\right) y'_M \right|, \left| y_M'^T \left(\frac{d}{n} J'\right) y'_M \right| &\leq 2 \sum_{i \in M} \sum_{j \in M : j \geq i} \frac{d}{n} 2^{-i} |S(u_i)| 2^{-j} |S(u_j)| \\ &\leq 2 \sum_{i \in M} \frac{1}{d} 2^{-2i} |S(u_i)| \sum_{j \in M : j \geq i} 2^{i-j} \\ &\leq \frac{4}{d} \|y'_M\|^2. \end{aligned}$$

The second inequality follows by noting that $|S(u_j)| \leq n/d^2 \ \forall \ j \in M$ ◀

▶ **Claim 14.**

$$|y_L^T A_s y_L| \leq \left(\frac{2\lambda}{5} + (3 \cdot 10^{12}) \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|y_L\|^2.$$

Proof. By triangle inequality,

$$\begin{aligned}|y_L^T A_s y_L| &= \left|\sum_{i,j\in L} (2^{-i}u_i^T)A_s(2^{-j}u_j)\right| \\ &\leq \left|\sum_{i,j\in L:i\leq j} (2^{-i}u_i)A_s(2^{-j}u_j)\right| + \left|\sum_{i,j\in L:i>j} (2^{-i}u_i)A_s(2^{-j}u_j)\right|.\end{aligned}$$

We bound each term using the first part of Lemma 12. We now clarify our choice of parameters to apply Lemma 12. For both terms, our choice is $r \leftarrow \max\{i \in L\}$, $\ell = r$, $u_i \leftarrow u_i$ if $i \in L$ and $u_i \leftarrow \bar{0}$ if $i \notin L$, $v_i = u_i$ for every $i \in [r]$, where $\bar{0}$ is the all-zeroes vector. We note that the conditions (I) and (II) of Lemma 12 are satisfied by this choice since every pair $S(u_i), S(u_j)$ is mutually disjoint and $|S(u_i)| > n/d^2$ for all $i \in L$. Consequently,

$$\begin{aligned}|y_L^T A_s y_L| &\leq 754 \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right) \sum_{i\in L} |S(u_i)|2^{-2i} + \left(\frac{\lambda}{5} + 2 \cdot 10^{12}\sqrt{d}\right) \sum_{j\in L} |S(u_j)|2^{-2j} \\ &\leq \left(\frac{2\lambda}{5} + (2 \cdot 10^{12} + 754) \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|y_L\|^2.\end{aligned}$$ ◀

▶ **Claim 15.**

$$|y_M^T A_s y_L| \leq 408 \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right) \|y_M\|^2 + \left(\frac{\lambda}{5} + (2 \cdot 10^{12}) \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|y_L\|^2.$$

Proof. By triangle inequality,

$$\begin{aligned}|y_M^T A_s y_L| &= \left|\sum_{i\in M, j\in L} (2^{-i}u_i^T)A_s(2^{-j}u_j)\right| \\ &\leq \left|\sum_{i\in M, j\in L:i\leq j} (2^{-i}u_i)A_s(2^{-j}u_j)\right| + \left|\sum_{i\in M, j\in L:i>j} (2^{-i}u_i)A_s(2^{-j}u_j)\right|.\end{aligned}$$

We bound the first and second terms by the first and second parts of Lemma 12 respectively. Let $\bar{0}$ be the all-zeroes vector. We now clarify our choice of parameters to apply Lemma 12. For the first term, our choice is $r \leftarrow \max\{i \in M\}$, $\ell \leftarrow \max\{i \in L\}$, $u_i \leftarrow u_i$ if $i \in M$ and $u_i \leftarrow \bar{0}$ if $i \notin M$, and $v_i \leftarrow u_i$ if $i \in L$ and $v_i \leftarrow \bar{0}$ if $i \notin L$. For the second term, our choice is $r \leftarrow \max\{i \in L\}$, $\ell \leftarrow \max\{i \in M\}$, $u_i \leftarrow u_i$ if $i \in L$ and $u_i \leftarrow \bar{0}$ if $i \notin L$, and $v_i \leftarrow u_i$ if $i \in M$ and $v_i \leftarrow \bar{0}$ if $i \notin M$. The conditions (I), (II) and (III) of Lemma 12 are satisfied for the respective choices since every pair $S(u_i), S(u_j)$ is mutually disjoint, $|S(u_i)| > n/d^2$ for all $i \in L$ and $|S(u_i)| > n/d^2 \geq |S(u_j)|$ for every $i \in L, j \in M$. Consequently,

$$\begin{aligned}|y_M^T A_s y_L| &\leq 377 \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right) \sum_{i\in M} |S(u_i)|2^{-2i} + \left(\frac{\lambda}{5} + 10^{12}\sqrt{d}\right) \sum_{j\in L} |S(u_j)|2^{-2j} \\ &\qquad +31 \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right) \left(\sum_{j\in L} |S(u_j)|2^{-2j} + \sum_{j\in M} |S(u_j)|2^{-2j}\right) \\ &\leq 408 \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right) \|y_M\|^2 + \left(\frac{\lambda}{5} + (10^{12} + 31) \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|y_L\|^2.\end{aligned}$$ ◀

From the above three claims, we have

$$\begin{aligned}|y^T A_s y| &\le \left(\lambda + 817 \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|y_M\|^2 + \\ &\qquad \left(\frac{4\lambda}{5} + 7 \cdot 10^{12} \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|y_L\|^2 \\ &\le \left(\lambda + 8 \cdot 10^{12} \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|y\|^2.\end{aligned}$$

Therefore, we have

$$\begin{aligned}|x^T A_s x| \le &\quad |y^T A_s y| \le \left(\lambda + 8 \cdot 10^{12} \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|y\|^2 \\ \le &\quad 4\left(\lambda + 8 \cdot 10^{12} \max\left(\sqrt{\lambda \log d}, \sqrt{d}\right)\right) \|x\|^2.\end{aligned}$$ ◀

We note that in the above proof, the multiplicative factor of 4 is a by-product of the discretization of x. This can be avoided if we do not discretize x straightaway, but instead "push" the discretization a little deeper into the proof. Indeed, we can see that the proof of Claim 13 where we bound $|y_M^T(A - (d/n)J)y_M|$ by $\lambda\|y_M\|^2$ does not require y_M to be a discretized vector. This is how we are able to prevent the multiplicative factor loss to obtain Theorem 2.

References

1 L. Addario-Berry and S. Griffiths. The spectrum of random lifts. Preprint arXiv:1012.4097, 2010.

2 N. Agarwal, K. Chandrasekaran, A. Kolla, and V. Madan. On the expansion of group–based lifts. Preprint arXiv:1311.3268, 2016. URL: https://arxiv.org/abs/1311.3268.

3 Y. Bilu and N. Linial. Lifts, discrepancy and nearly optimal spectral gap. *Combinatorica*, 26(5):495–519, 2006.

4 C. Bordenave. A new proof of friedman's second eigenvalue theorem and its extension to random lifts. Preprint arXiv:1502.04482, 2015.

5 K. Chandrasekaran and A. Velingker. Shift lifts preserving ramanujan property. *Linear Algebra and its Applications*, 529:199–214, 2017.

6 F. Chung. Diameters and eigenvalues. *Journal of the American Mathematical Society*, 2(2):187–196, 1989.

7 M. Cohen. Ramanujan graphs in polynomial time. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 276–281, 2016.

8 R. Feng, J. Kwak, and J. Lee. Characteristic polynomials of graph coverings. *Bull. Austal. Math. Soc.*, 69:133–136, 2004.

9 J. Friedman. Relative expanders or weakly relatively ramanujan graphs. *Duke Math. J*, 118:2003, 2003.

10 J. Friedman. A proof of alon's second eiganvalue conjecture and related problems. *Mem. Amer. Math,Soc*, 195(910), 2008.

11 J. Friedman, J. Kahn, and E. Szemerédi. On the second eigenvalue of random regular graphs. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC'89, pages 587–598, 1989.

12 J. Friedman and D.-E. Kohler. The Relativized Second Eigenvalue Conjecture of Alon. Preprint arXiv:1403.3462, 2014.

13 J. Friedman, R. Murty, and J. Tillich. Spectral estimates for abelian cayley graphs. *J. Comb. Theory Ser. B*, 96(1):111–121, 2006.

14 Y. Greenberg. On the spectrum of graphs and their universal coverings. Ph.D Thesis, 1995.

15 Chris Hall, Doron Puder, and William F. Sawin. Ramanujan coverings of graphs. In *Proceedings of the Forty-eighth Annual ACM Symposium on Theory of Computing*, STOC'16, pages 533–541, 2016.

16 S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bull. Amer. Math. Soc*, 43(4):439–561, 2006.

17 N. Linial and D. Puder. Word maps and spectra of random graph lifts. *Random Struct. Algorithms*, 37(1)):100–135, 2010.

18 E. Lubetzky, B. Sudakov, and V. Vu. Spectra of lifted ramanujan graphs. *Advances in Mathematics*, 227:1612–1645, 2011.

19 A. Lubotzky, R. Phillips, and P. Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

20 A. Makelov. Expansion in lifts of graphs, 2015. Undergraduate Thesis, Harvard University.

21 A. Marcus, D. Spielman, and N. Srivastava. Interlacing families i: Ramanujan graphs of all degrees. In Proceedings, FOCS 2013, 2013.

22 A. Marcus, D. Spielman, and N. Srivastava. Interlacing families iv: Bipartite ramanujan graphs of all sizes. In *IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 1358–1377, 2015.

23 G. Margulis. Explicit group-theoretic constructions of combinatorial schemes and their applications in the construction of expanders and concentrators. *Probl. Inf. Transm*, 24(1):39–46, 1988.

24 H. Mizuno and I. Sato. Characteristic polynomials of some graph coverings. *Discrete Mathematics*, 142:295–298, 1995.

25 A. Nilli. On the second eigenvalue of a graph. *Discrete Math*, 91(2):207–210, 1991.

26 M. Pinsker. On the complexity of a concentrator. *7th International Teletraffic Conference*, pages 318/1–318/4, 1973.

27 D. Puder. Expansion of random graphs: New proofs, new results. Preprint arXiv:1212.5216, 2013.

28 P. Sarnak. What is an expander? *Notices Amer. Math. Soc*, 51(7):762–763, 2006.

Efficient Removal Lemmas for Matrices

Noga Alon[*1] **and Omri Ben-Eliezer**[2]

1 **Departments of Mathematics and Computer Science, Tel Aviv University, Tel Aviv, Israel**
nogaa@tau.ac.il

2 **Department of Computer Science, Tel Aviv University, Tel Aviv, Israel**
omrib@mail.tau.ac.il

Abstract

The authors and Fischer recently proved that any hereditary property of two-dimensional matrices (where the row and column order is not ignored) over a finite alphabet is testable with a constant number of queries, by establishing the following (ordered) matrix removal lemma: For any finite alphabet Σ, any hereditary property $\mathcal{P}$ of matrices over Σ, and any $\epsilon > 0$, there exists $f_{\mathcal{P}}(\epsilon)$ such that for any matrix M over Σ that is ϵ-far from satisfying $\mathcal{P}$, most of the $f_{\mathcal{P}}(\epsilon) \times f_{\mathcal{P}}(\epsilon)$ submatrices of M do not satisfy $\mathcal{P}$. Here being ϵ-far from $\mathcal{P}$ means that one needs to modify at least an ϵ-fraction of the entries of M to make it satisfy $\mathcal{P}$.

However, in the above general removal lemma, $f_{\mathcal{P}}(\epsilon)$ grows very fast as a function of ϵ^{-1}, even when $\mathcal{P}$ is characterized by a single forbidden submatrix. In this work we establish much more efficient removal lemmas for several special cases of the above problem. In particular, we show the following: For any fixed $s \times t$ binary matrix A and any $\epsilon > 0$ there exists $\delta > 0$ polynomial in ϵ, such that for any binary matrix M in which less than a δ-fraction of the $s \times t$ submatrices are equal to A, there exists a set of less than an ϵ-fraction of the entries of M that intersects every A-copy in M.

We generalize the work of Alon, Fischer and Newman [SICOMP'07] and make progress towards proving one of their conjectures. The proofs combine their efficient conditional regularity lemma for matrices with additional combinatorial and probabilistic ideas.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Property Testing, Removal Lemma, Matrix Regularity Lemma, Binary Matrix

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.25

1 Introduction

Removal lemmas are structural combinatorial results that relate the *density* of "forbidden" substructures in a given large structure S with the *distance* of S from not containing any of the forbidden substructures, stating that if S contains a small number of forbidden substructures, then one can make S free of such substructures by making only a small number of modifications in it. Removal lemmas are closely related to many problems in Extremal Combinatorics, and have direct implications in Property Testing and other areas of Mathematics and Computer Science, such as Number Theory and Discrete Geometry.

The first known removal lemma has been the celebrated (non-induced) *graph removal lemma*, established by Rusza and Szemerédi [24] (see also [3, 4]). This fundamental result in

[*] Research supported in part by a USA-Israeli BSF grant 2012/107, by an ISF grant 620/13 and by the Israeli I-Core program.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 25; pp. 25:1–25:18
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Graph Theory states that for any fixed graph H on h vertices and any $\epsilon > 0$ there exists $\delta > 0$, such that for any graph G on n vertices that contains at least ϵn^2 copies of H that are pairwise edge-disjoint, the total number of H-copies in G is at least δn^h. Many extensions and strengthenings of the graph removal lemma have been obtained, as is described in more detail in Section 2.

In this work, we consider removal lemmas for two-dimensional *matrices* (with row and column order) over a finite alphabet. For simplicity, the results are generally stated for square matrices, but are easily generalizable to non-square matrices. Some of the results also hold for matrices in more than two dimensions.

The notation below is given for two-dimensional matrices, but carries over naturally to other combinatorial structures, such as graphs and multi-dimensional matrices.

An $m \times n$ matrix M over the alphabet Γ is viewed here as a function $M : [m] \times [n] \to \Gamma$, and the row and column order is dictated by the natural order on their indices. Any matrix that can be obtained from a matrix M by deleting some of its rows and columns (while preserving the row and column order) is considered a *submatrix* of M. We say that M is *binary* if the alphabet is $\Gamma = \{0, 1\}$ and *ternary* if $\Gamma = \{0, 1, 2\}$. A *matrix property* $\mathcal{P}$ over Γ is simply a collection of matrices $M : [m] \times [n] \to \Gamma$. A matrix is *$\epsilon$-far* from $\mathcal{P}$ if one needs to change at least an ϵ-fraction of its entries to get a matrix that satisfies $\mathcal{P}$. A property $\mathcal{P}$ is *hereditary* if it is closed under taking submatrices, that is, if $M \in \mathcal{P}$ then any submatrix M' of M satisfies $M' \in \mathcal{P}$. For any family $\mathcal{F}$ of matrices over Γ, the property of $\mathcal{F}$-freeness, denoted by $\mathcal{P}_{\mathcal{F}}$, consists of all matrices over Γ that do not contain a submatrix from $\mathcal{F}$. Observe that $\mathcal{P}$ is hereditary if and only if it is characterized by some family $\mathcal{F}$ of forbidden submatrices, i.e. $\mathcal{P} = \mathcal{P}_{\mathcal{F}}$.

While the investigation of graph removal lemmas has been quite extensive, as described in Section 2 below, the first known removal lemma for ordered graph-like two-dimensional structures, and specifically for (row and column ordered) matrices, was only obtained very recently by the authors and Fischer [2].

▶ **Theorem 1** ([2]). *Fix a finite alphabet Γ. For any hereditary property $\mathcal{P}$ of matrices over Γ and any $\epsilon > 0$ there exists $f_{\mathcal{P}}(\epsilon)$ satisfying the following. If a matrix M is ϵ-far from $\mathcal{P}$ then at least a $2/3$-fraction of the $f_{\mathcal{P}}(\epsilon) \times f_{\mathcal{P}}(\epsilon)$ submatrices of M do not satisfy $\mathcal{P}$.*

However, even when $\mathcal{P}$ is characterized by a single forbidden submatrix, the upper bound on $f_{\mathcal{P}}(\epsilon)$ guaranteed by the removal lemma in [2] is very large; in fact, it is at least as large as a wowzer (tower of towers) type function of ϵ. On the other hand, a lower bound of Fischer and Rozenberg [15] implies that one cannot hope for a polynomial dependence of $f_{\mathcal{P}}(\epsilon)$ in ϵ^{-1} in general (for the non-binary case), even when $\mathcal{P}$ is characterized by a single forbidden submatrix.

Thus, it is natural to ask for which hereditary matrix properties $\mathcal{P}$ there exist removal lemmas with more reasonable upper bounds on $f_{\mathcal{P}}(\epsilon)$, and specifically, to identify large families of properties $\mathcal{P}$ for which $f_{\mathcal{P}}(\epsilon)$ is *polynomial* in ϵ^{-1}. In this work we focus on this question, mainly for matrices over a binary alphabet.

A natural motivation for the investigation of removal lemmas comes from *property testing*. This active field of study in computer science, initiated by Rubinfeld and Sudan [23] (see [18] for the graph case), is dedicated to finding fast algorithms to distinguish between objects that satisfy a certain property and objects that are far from satisfying this property; these algorithms are called testers. An *ϵ-tester* for a matrix property $\mathcal{P}$ is a (probabilistic) algorithm that is given query access to the entries of the input matrix M, and is required to distinguish, with error probability at most $1/3$, between the case that M satisfies $\mathcal{P}$ and the case that M is ϵ-far from $\mathcal{P}$. If the tester always answers correctly when M satisfies $\mathcal{P}$, we say that the tester has a one-sided error. We say that $\mathcal{P}$ is testable if there is a one-sided error tester for $\mathcal{P}$ that makes a constant number of queries (that depends only on $\mathcal{P}$ and ϵ but not on the size of the input). Furthermore, $\mathcal{P}$ is *easily testable* if the number of queries is polynomial in ϵ^{-1}. Clearly, any hereditary property of matrices is testable by Theorem 1, while any property $\mathcal{P}$ for which $f_{\mathcal{P}}(\epsilon)$ is shown to be polynomial in ϵ^{-1} is easily testable.

1.1 Background and main results

The results here are stated and proved for square $n \times n$ matrices, but can be generalized to non-square matrices in a straightforward manner. Our first main result is an efficient *weak removal lemma* for binary matrices.

▶ **Theorem 2.** *If an $n \times n$ binary matrix M contains ϵn^2 pairwise-disjoint copies of an $s \times t$ binary matrix A, then the total number of A-copies in M is at least δn^{s+t}, where δ^{-1} is polynomial in ϵ^{-1}.*

Here a set of pairwise-disjoint A-copies in M is a set of $s \times t$ submatrices of M, all equal to A, such that any entry of M is contained in at most one of the submatrices.

Theorem 2 is an analogue for binary matrices of the non-induced graph removal lemma. However, in the graph removal lemma, δ^{-1} is not polynomial in ϵ^{-1} in general, in contrast to the situation in Theorem 2.

Alon, Fischer and Newman [5] proved an efficient induced removal lemma for a certain type of finite families $\mathcal{F}$ of binary matrices. A family $\mathcal{F}$ of matrices, or equivalently, a hereditary matrix property $\mathcal{P}_{\mathcal{F}}$, is *closed under row (column) permutations* if for any $A \in \mathcal{F}$, any matrix created by permuting the rows (columns respectively) of A is in $\mathcal{F}$. $\mathcal{F}$ is *closed under permutations* if it is closed under row permutations and under column permutations.

▶ **Theorem 3** ([5]). *Let $\mathcal{F}$ be a finite family of binary matrices that is closed under permutations. For any $\epsilon > 0$ there exists $\delta > 0$, where δ^{-1} is polynomial in ϵ^{-1}, such that any $n \times n$ binary matrix that is ϵ-far from $\mathcal{F}$-freeness contains δn^{s+t} copies of some $s \times t$ matrix $A \in \mathcal{F}$.*

The main consequence of Theorem 3 is an efficient induced removal lemma for bipartite graphs. Indeed, when representing a bipartite graph by its (bi-)adjacency matrix, a forbidden subgraph H is represented by the family $\mathcal{F}$ of all matrices that correspond to bipartite graphs isomorphic to H. Note that $\mathcal{F}$ is indeed closed under permutations in this case. Thus, any hereditary bipartite graph property characterized by a finite set of forbidden induced subgraphs is easily testable.

The problem of understanding whether the statement of Theorem 3 holds for *any* finite family $\mathcal{F}$ of binary matrices, was raised in [5] and is still open. Only recently in [2] it was shown that the statement holds if we ignore the polynomial dependence, as stated in Theorem 1.

▶ **Problem 4.** *Is it true that for any fixed finite family $\mathcal{F}$ of binary matrices and any $\epsilon > 0$, there exists $\delta > 0$ with δ^{-1} polynomial in ϵ^{-1}, such that any $n \times n$ binary matrix M that is ϵ-far from $\mathcal{F}$-freeness contains δn^{s+t} copies of some $s \times t$ matrix $A \in \mathcal{F}$?*

Theorem 2 implies that to settle Problem 4 it is enough to show the following. Fix a finite family $\mathcal{F}$ of binary matrices. Then for any $\epsilon > 0$ there exists $\tau > 0$, with τ^{-1} polynomial in ϵ^{-1}, such that any $n \times n$ binary matrix that is ϵ-far from $\mathcal{F}$-freeness contains τn^2 pairwise disjoint copies of matrices from $\mathcal{F}$.

Our second main result makes progress towards solving Problem 4 by generalizing the statement of Theorem 3 to any family $\mathcal{F}$ of binary matrices that is closed under row (or column) permutations. From now on we only state the results for families that are closed under row permutations, but analogous results hold for families closed under column permutations.

▶ **Theorem 5.** *Let $\mathcal{F}$ be a finite family of binary matrices that is closed under row permutations. For any $\epsilon > 0$ there exists $\delta > 0$, where δ^{-1} is polynomial in ϵ^{-1}, such that any $n \times n$ binary matrix that is ϵ-far from $\mathcal{F}$-freeness contains δn^{s+t} copies of some $s \times t$ matrix $A \in \mathcal{F}$.*

▶ **Corollary 6.** *Any hereditary property of binary matrices that is characterized by a finite forbidden family closed under row permutations is easily testable.*

Our proof of Theorem 5 is somewhat simpler than the original proof of Theorem 3. One of the main tools in the proofs of Theorems 2 and 5 is an efficient conditional regularity lemma for matrices developed in [5] (see also [20]). In the proof of Theorem 5 we only use a simpler form of the lemma, which is also easier to prove. The statement of the lemma and the proofs of Theorems 2, 5 appear in Section 4.

Besides the above two main results, we also describe a simpler variant of the construction of Fischer and Rozenberg [15], showing that for ternary matrices, the dependence between the parameters is not polynomial in general. We further suggest a way to tackle the weak removal lemma (i.e. the analogue of Theorem 2, without the polynomial dependence) in high dimensional matrices over arbitrary alphabets, by reducing it to an equivalent problem that looks more accessible. For more details, see Section 5.

2 Related work

Removal lemmas have been studied extensively in the context of graphs. The non-induced graph removal lemma (which was stated in the beginning of Section 1) has been one of the first applications of the celebrated Szemerédi graph regularity lemma [26]. The *induced graph removal lemma*, established in [4] by proving a stronger version of the graph regularity lemma, is a similar result considering induced subgraphs. It states that for any finite family $\mathcal{F}$ of graphs and any $\epsilon > 0$ there exists $\delta = \delta(\mathcal{F}, \epsilon) > 0$ with the following property. If an n-vertex graph G is ϵ-far from $\mathcal{F}$-freeness, then it contains at least $\delta n^{v(F)}$ induced copies of some $F \in \mathcal{F}$. Here $v(F)$ denotes the number of vertices in F, and G is said to be (induced) $\mathcal{F}$-free if no induced subgraph of G is isomorphic to a graph from $\mathcal{F}$.

The induced graph removal lemma was later extended to infinite families [9], stating the following. For any finite or infinite family $\mathcal{F}$ of graphs and any $\epsilon > 0$ there exists $f_{\mathcal{F}}(\epsilon)$ with the following property. If an n-vertex graph G is ϵ-far from $\mathcal{F}$-freeness, then with probability at least $2/3$, a random induced subgraph of G on $f_{\mathcal{F}}(\epsilon)$ vertices contains a graph from $\mathcal{F}$. Note that when $\mathcal{F}$ is finite, the statement of the infinite induced removal lemma is indeed equivalent to that of the finite version of the induced removal lemma.

The graph removal lemma was also extended to hypergraphs [19, 22, 21, 27]. See [13] for many more useful variants, quantitative strengthenings and extensions of the graph removal lemma.

Very recently, the authors and Fischer [2] generalized the (finite and infinite) induced graph removal lemma by obtaining an *order-preserving* version of it, and also showed that the same type of proof can be used to obtain a removal lemma for two-dimensional *matrices* (with row and column order) over a finite alphabet; this it Theorem 1 above.

However, even for the non-induced graph removal lemma where the forbidden subgraph is a triangle, the best known general upper bound for δ^{-1} in terms of ϵ^{-1} is of tower-type [16, 12].

On the other hand, the best known lower bound for the dependence is super-polynomial but sub-exponential, and builds on a construction of Behrend [10]. See [1] for more details. Understanding the “right” dependence of δ^{-1} in ϵ^{-1}, even for the simple case where the forbidden graph H is a triangle, is considered an important and difficult open problem.

In view of the above discussion, a lot of effort has been dedicated to the problem of characterizing the hereditary graph properties $\mathcal{P}$ for which $f_{\mathcal{P}}(\epsilon)$ is polynomial in ϵ^{-1}, i.e., the easily testable graph properties. See the recent work of Gishboliner and Shapira [17]; for other previous works on this subject, see, e.g., [1, 8, 6]. Our work also falls under this category, but for (ordered) matrices instead of graphs; it is the first work of this type for *ordered* two-dimensional graph-like structures.

We finish by mentioning several other relevant removal lemma type results. Removal lemmas for vectors (i.e. one dimensional matrices where the order is important) are generally easier to obtain; in particular, a removal lemma for vectors over a fixed finite alphabet can be derived from a removal lemma for regular languages proved in [7]. A removal lemma for partially ordered sets with a grid-like structure, which can be seen as a generalization of the removal lemma for vectors, can be deduced from a result of Fischer and Newman in [14], where they mention that this problem for submatrices is more complicated and not understood. Recently, Ben-Eliezer, Korman and Reichman [11] obtained a removal lemma for patterns in multi-dimensional matrices. A pattern must be taken from *consecutive* locations, whereas in our case the rows and columns of a submatrix need not be consecutive. The case of patterns behaves very differently than that of submatrices, and in particular, in the removal lemma for patterns the parameters are linearly related (for any alphabet size) unlike the case of submatrices (in which, for alphabets of 3 letters or more, the relation cannot be polynomial).

3 Notation

Here we give some more notation that will be useful throughout the rest of the paper. We give the notation for rows but the notation for columns is equivalent. Let $M : [m] \times [n] \to \Gamma$ be an $m \times n$ matrix. For two rows in M whose indices in I are $r < r'$, we say that row r is *smaller* than row r' and row r' is *larger* than row r. The *predecessor* of row r in M is the largest row $\bar{r}$ in M smaller than r. In this case we say that r is the *successor* of $\bar{r}$.

Let S be the submatrix of M on $\{r_1, \dots, r_s\} \times \{c_1, \dots, c_t\}$ where $r_1 < \dots < r_s$ and $c_1 < \dots < c_t$. For $i = 1, \dots, s$, the *i-row-index* of S in M is r_i; For two submatrices S, S' of the same dimensions and with i-row-indices r_i, r'_i respectively we say that S is *i-row-smaller* than S' if $r_i < r'_i$ and *i-row-bigger* if $r_i > r'_i$.

Let $X = \{x_1, \dots, x_{s-1}\} \subseteq [m]$ with $0 < x_1 < \dots < x_{s-1} < m$ and $Y = \{y_1, \dots, y_{t-1}\} \subseteq [n]$ with $0 < y_1 < \dots < y_{t-1} < n$ be subsets of indices. The submatrix S is *row-separated* by X if $r_i \le x_i < r_{i+1}$ for any $i = 1, \dots, s-1$, *column-separated* by Y if $c_j \le y_j < c_{j+1}$ for any $j = 1, \dots, t-1$ and *separated* by $X \times Y$ if it is row separated by X and column separated by Y. The elements of X, Y are called *row separators*, *column separators* respectively.

3.1 Folding and unfoldable matrices

A matrix is *unfoldable* if no two neighboring rows in it are equal and no two neighboring columns in it are equal. The *folding* of a matrix A is the unique matrix $\tilde{A}$ generated from A by deleting any row of A that is equal to its predecessor, and then deleting any column of the resulting matrix that is equal to its predecessor. Note that $\tilde{A}$ is unfoldable.

▶ **Lemma 7.** *Fix an $s \times t$ matrix A and let $\tilde{A}$ be its $s' \times t'$ folding. For any $\epsilon > 0$ there exist $n_0, \delta > 0$, where n_0 and δ^{-1} are polynomial in ϵ^{-1}, such that for any $n \geq n_0$, any $n \times n$ matrix M that contains $\epsilon n^{s'+t'}$ copies of $\tilde{A}$ also contains δn^{s+t} copies of A.*

Lemma 7 implies that generally, to prove removal lemma type results for finite families, it is enough to only consider families of unfoldable matrices. The proof follows immediately from the following lemma.

▶ **Lemma 8.** *Let A be an $s \times t$ fixed matrix and let A' be an $s' \times t$ matrix created from A by deleting rows that are equal to their predecessors in A. Then for any $\epsilon > 0$ there exist $n_1 = n_1(A, \epsilon) > 0$ and $\tau = \tau(A, \epsilon) > 0$, where n_1 and τ^{-1} are polynomial in ϵ^{-1}, such that for any $n \geq n_1$, any $n \times n$ matrix M that contains $\epsilon n^{s'+t}$ copies of A' also contains τn^{s+t} copies of A.*

Proof of Lemma 8. Let T be the family of all $n \times t$ submatrices S of M containing at least $\epsilon n^{s'}/2$ copies of A'. Any $S \in T$ has $\binom{n}{s'} \leq n^{s'}$ $s' \times t$ submatrices, so the number of A' copies in submatrices from T is at most $|T|n^{s'}$. On the other hand, there are $\binom{n}{t} \leq n^t$ $n \times t$ submatrices of M so the number of A' copies in $n \times t$ submatrices not in T is less than $\epsilon n^{s'+t}/2$. Hence the total number of A' copies in submatrices from T is at least $\epsilon n^{s'+t}/2$, implying that $|T| \geq \epsilon n^t/2$.

Observe that any $S \in T$ contains a collection $\mathcal{A}(S)$ of $\epsilon n/2s'$ pairwise disjoint copies of A'. To show this, we follow a greedy approach, starting with a collection $\mathcal{B}$ of all A'-copies in S and with empty $\mathcal{A}$. As long as $\mathcal{B}$ is not empty, we arbitrarily choose a copy $C \in \mathcal{B}$ of A', add C to $\mathcal{A}$ and delete all A'-copies intersecting C (including itself) from $\mathcal{B}$. In each step, the number of deleted copies is at most $s'n^{s'-1}$, so the number of steps is at least $\epsilon n^{s'}/2s'n^{s'-1} = \epsilon n/2s'$.

Let $\delta = \epsilon/5ss'$ and take $S \in T$. Assuming that n is large enough, pick disjoint collections $\mathcal{A}_1, \ldots, \mathcal{A}_s \subseteq \mathcal{A}(S)$, each of size at least δn, so that all A'-copies in $\mathcal{A}_i$ are i-row-smaller than all A'-copies in $\mathcal{A}_{i+1}$ for any $1 \leq i \leq s-1$. Then there are $\delta^s n^s$ copies of A in S: Each $s \times t$ submatrix of S whose i-th row is taken as the i-th row of a matrix from $\mathcal{A}_i$ is equal to A. Therefore, the total number of A-copies in M is at least $|T|\delta^s n^s \geq \epsilon \delta^s n^{s+t}/2$, as desired. ◀

4 Proofs for the binary case

This section is dedicated to the proof of our main results in the binary domain: Theorem 2 and Theorem 5. As a general remark for the proofs in this section, We may and will assume that a square matrix M is sufficiently large (given $\epsilon > 0$), by which we mean that M is an $n \times n$ matrix with $n \geq n_0$ for a suitable $n_0 > 0$ that is polynomial in ϵ^{-1}.

One of the main tools in the proofs of this section is a conditional regularity lemma for matrices due to Alon, Fischer and Newman [5]. We describe a simpler version of the lemma (this is Lemma 9 below) along with another useful result from their paper (Lemma 10 below). Combining these results together yields the original version of the conditional regularity lemma used in the original proof of Theorem 3 in [5]. It is worth to note that even though Theorem 5 generalizes Theorem 3, for its proof we only need the simpler Lemma 9 and not the original regularity lemma, whose proof requires significantly more work. Lemma 10 is only used in the proof of Theorem 2.

We start with some definitions. A (δ, r)-*row-clustering* of an $n \times n$ matrix M is a partition of the set of rows of M into $r + 1$ *clusters* $R_0, \ldots, R_r$ such that the *error cluster* R_0 satisfies $|R_0| \leq \delta n$ and for any $i = 1, \ldots, r$, every two rows in R_i differ in at most δn entries. That is, for every $e, e' \in R_i$, one can make row e equal to e' by modifying at most δn entries.

A (δ, r)-*column-clustering* is defined analogously on the set of columns of M. The first conditional regularity lemma states the following.

▶ **Lemma 9** ([5]). *Let k be a fixed positive integer and let $\delta > 0$ be a small real. For every $n \times n$ binary matrix M with $n > (k/\delta)^{O(k)}$, either M admits (δ, r)-clusterings for both the rows and the columns with $r \leq (k/\delta)^{O(k)}$, or for every $k \times k$ binary matrix A, at least a $(\delta/k)^{O(k^2)}$ fraction of the $k \times k$ submatrices of M are copies of A.*

Let R be a set of rows and let C be a set of columns in an $n \times n$ matrix M. The *block* $R \times C$ is the submatrix of M on $R \times C$. A block B is δ-*homogeneous* with *value* b if there exists $b \in \{0, 1\}$ such that at least a $1 - \delta$ fraction of the entries of B are equal to b. A (δ, r)-*partition* of M is a couple $(\mathcal{R}, \mathcal{C})$ where $\mathcal{R} = \{R_1, \ldots, R_r\}$ is a partition of the set of rows and $\mathcal{C} = \{C_1, \ldots, C_r\}$ is a partition of the set of columns of M, such that all but a δ-fraction of the entries of M lie in blocks $R_i \times C_j$ that are δ-homogeneous. The second result that we need from [5], relating clusterings and partitions of a matrix, is as follows.

▶ **Lemma 10** ([5]). *Let $\delta > 0$. If a square binary matrix M has $(\delta^2/16, r)$-clusterings $\mathcal{R}, \mathcal{C}$ of the rows and the columns respectively then $(\mathcal{R}, \mathcal{C})$ is a $(\delta, r + 1)$-partition of M.*

For the proofs of the above lemmas see [5]. We continue to the proof of Theorem 2. The following lemma is a crucial part of the proof.

▶ **Lemma 11.** *Fix an $s \times t$ matrix A. For any $\epsilon > 0$ there exists $\tau > 0$, where τ^{-1} is polynomial in ϵ^{-1}, such that any $n \times n$ matrix M containing ϵn^2 pairwise-disjoint copies of A either contains τn^{s+t} copies of any $s \times t$ matrix, or there exist subsets of indices X, Y of sizes $s - 1, t - 1$ respectively such that M contains τn^2 pairwise disjoint copies of A that are separated by $X \times Y$.*

Before providing the full proof of Lemma 11, we present a sketch of the proof. Clearly, whenever we apply Lemma 9 throughout the proof, we may assume that the outcome is that M has suitable row and column clusterings, as the other possible outcome of Lemma 9 finishes the proof immediately. The main idea of the proof is to gradually find row separators, and then column separators, while maintaining a large set of pairwise disjoint copies of A that conform to these separators. This is done inductively (first for the rows, and then for the columns). The inductive step is described in what follows.

Assume we currently have $j - 1 \geq 0$ row-separators, and a set $\mathcal{A}$ of many pairwise disjoint A-copies that have their first j rows separated by these row-separators. We take a clustering of the rows of M, and consider a cluster in which many rows are "good", in the sense that they contain the j-th row of many of the disjoint A-copies from $\mathcal{A}$. We put our j-th separator as the medial row among the good rows. Next, we consider a matching of pairs (r_1, r_2) of good rows, where in each such pair r_1 lies before the j-th separator and r_2 lies after the j-th separator. Observe that all good rows lie after the $(j - 1)$-th separator.

If we take all pairwise-disjoint A-copies from $\mathcal{A}$ whose j-th row is r_2, and "shift" their j-th row to be r_1, then most of them will still be A-copies (as rows r_1 and r_2 are very similar, since they are in the same row cluster). This process creates a set $\mathcal{A}'$ of many pairwise disjoint A-copies whose i-th row lies between separators $i - 1$ and i for any $i \leq j$, and the $(j + 1)$-th row lies after separator j. This finishes the inductive step.

We now continue to the full proof of Lemma 11.

Proof of Lemma 11. Let $\epsilon > 0$ and let M be a large enough $n \times n$ binary matrix containing a collection U_0 of ϵn^2 pairwise disjoint A-copies.

We prove the following claim by induction on i, for $i = 0, 1, \ldots, s-1$: there exist τ_i, δ_i with $\tau_i^{-1}, \delta_i^{-1}$ polynomial in ϵ^{-1} such that either M contains $\tau_i n^{s+t}$ copies of any $s \times t$ matrix or there exist $0 = x_0 < x_1 < \ldots < x_i$ and a set U_i of $\delta_i n^2$ pairwise disjoint A-copies in M whose j-th row is bigger then x_{j-1} and no bigger than x_j for any $1 \leq j \leq i$, and the $(i+1)$-th row is bigger than x_i. The base case $i = 0$ is trivial with $\delta_0 = \epsilon$. Suppose now that $i \geq 1$ and that $x_0, \ldots, x_{i-1}$, δ_{i-1} and U_{i-1} are already determined. Applying Lemma 9 on M with parameters $k = \max\{s, t\}$ and $\delta_{i-1}/4$, either M contains $\tau_i n^{s+t}$ copies of any $s \times t$ matrix with τ_i^{-1} polynomial in ϵ^{-1} and we are done, or M has a $(\delta_{i-1}/4, r_i)$-row-clustering $\mathcal{R}_i$ of M for r_i polynomial in δ_{i-1}^{-1} and so in ϵ^{-1}. The number of rows of M that contain the i-th row of at least $\delta_{i-1}n/2$ of the A-copies in U_{i-1} is at least $\delta_{i-1}n/2$, since the number of A-copies in U_{i-1} whose i-th row is not taken from such a row of M is less that $n \cdot \delta_{i-1}n/2 = \delta_{i-1}n^2/2$. Let R_i be a row cluster that contains at least $\delta_{i-1}n/2r_i$ such rows. Note that all of these rows are bigger than x_{i-1}. Take subclusters R_i^1, R_i^2 of R_i, each containing at least $\lfloor \delta_{i-1}n/4r_i \rfloor \geq \delta_{i-1}n/5r_i$ such rows (the inequality holds for n large enough) where each row in R_i^1 is smaller than each row in R_i^2. Take x_i to be the row index of the biggest row in R_i^1.

Take arbitrarily $\delta_{i-1}n/5r_i$ couples of rows (r, r') where $r \in R_i^2$ and $r' \in R_i^1$ and every row participates in at most one couple. Let (r, r') be such a couple. There exist $\delta_{i-1}n/2$ $s \times t$ submatrices of M that are A-copies from U_{i-1} and whose i-th row is r. Moreover, for any $j < i$ the j-th row of each of these submatrices lies between x_{j-1} (non-inclusive) and x_j (inclusive). Since r and r' differ in at most $\delta_{i-1}n/4$ entries, there are at least $\delta_{i-1}n/4$ such submatrices T that satisfy the following: If we modify T by taking its i-th row to be r' instead of r, T remains an A-copy. Moreover, after the modification, the i-th row of T is in R_i^1 and is therefore no bigger than x_i, whereas the $(i+1)$-th row of T is bigger than the i-th row of T before the modification which is bigger than x_i, as needed. For every couple (r, r') we can produce $\delta_{i-1}n/4$ pairwise disjoint copies of A whose j-th row is between x_{j-1} and x_j for any $j \geq i$ and the $(i+1)$-th row is after x_i. There are $\delta_{i-1}n/5r_i$ such couples (r, r'), and in total we get a set U_i of $\delta_i n^2$ copies of A with the desired structure for $\delta_i = \delta_{i-1}^2/20r_i$ where δ_i^{-1} is polynomial in δ_{i-1}^{-1} and so in ϵ^{-1}. Note that the copies in U_i are pairwise disjoint. In the end of the process there is a set $U = U_s$ of $\delta_s n^2$ pairwise disjoint copies of A whose rows are separated by $X = \{x_1, \ldots, x_{s-1}\}$. A feature that is useful in what follows is that each copy in U has exactly the same set of columns (as a submatrix of M) as one of the original copies of U_0.

Now we apply the same process as above but in columns instead of rows, starting with the $\delta_s n^2$ copies in U. In the end of the process, we obtain that for some $\hat{\tau}_t, \hat{\delta}_t$ such that $\hat{\tau}_t^{-1}$ and $\hat{\delta}_t^{-1}$ are polynomial in δ_s^{-1} and so in ϵ^{-1}, either M contains $\hat{\tau}_t n^{s+t}$ copies of any $s \times t$ matrix, or there exists a set $\hat{U}$ of $\hat{\delta}_t n^2$ pairwise disjoint copies of A whose columns are separated by a set of indices Y of size $t - 1$. Moreover, by the above feature, each of the copies in $\hat{U}$ has the same set of rows as some copy of A from U, so each copy has its rows separated by X. Hence $X \times Y$ separates all copies in $\hat{U}$. Taking $\tau = \min\{\hat{\tau}_t, \hat{\delta}_t\}$ finishes the proof. ◀

Next we show how Theorem 2 follows from Lemma 11. The idea of the proof is to show, using Lemmas 10 and 11, that there is a partition of M with blocks $R_i \times C_j$ (for $1 \leq i \leq s$, $1 \leq j \leq t$) satisfying the following.

- All row clusters R_i and all column clusters C_j are large enough.
- All rows of R_i (C_j) lie before all rows (columns) of R_{i+1} (C_{j+1} respectively) for any i and j.
- $R_i \times C_j$ is almost homogeneous, and its "popular" value is A_{ij}.

Using these properties, it is easy to conclude that M contains many A-copies.

We now complete the proof of Theorem 2.

Proof of Theorem 2. Let A be an $s \times t$ binary matrix and let $k = \max\{s, t\}$. Let $\epsilon > 0$ and let M be a large enough $n \times n$ binary matrix that contains ϵn^2 pairwise disjoint A-copies. Lemma 11 implies that either M contains τn^{s+t} copies of A where τ^{-1} is polynomial in ϵ^{-1} (in this case we are done), or M contains at least τn^2 pairwise disjoint copies of A separated by $X \times Y$ for suitable index subsets X, Y. By Lemma 9 we get that either M has $(\tau^2/128, r)$-clusterings of the rows and the columns where r is polynomial in τ^{-1} and so in ϵ^{-1}, or at least a $\zeta = (\tau^2/128k)^{O(k^2)}$ fraction of the $s \times t$ submatrices are A; in the second case we are done. Suppose then that M has $(\tau^2/128, r)$-clusterings $\mathcal{R}, \mathcal{C}$ of the rows, columns respectively. The next step is to create refinements of the clusterings. Write the elements of X as $x_1 < \ldots < x_{s-1}$ and let $x_0 = 0, x_s = n$. Partition each $R \in \mathcal{R}$ into s parts where the i-th part for $i = 1, \ldots, s$ consists of all rows in R with index at least x_{i-1} and less than x_i. Each such part is also a $\tau^2/128$-cluster. Now separate each $C \in \mathcal{C}$ into t parts in a similar fashion. This creates $(\tau^2/128, (r+1)k)$-clusterings $\mathcal{R}', \mathcal{C}'$ of the rows and the columns respectively (where some of the clusters might be empty). By Lemma 10, $\mathcal{P} = (\mathcal{R}', \mathcal{C}')$ is a $(\tau/4, r')$-partition of M where $r' = (r+1)k + 1$, and each block of the partition has all of its entries between two neighboring row separators from X and between two neighboring column separators from Y.

There are at most $\tau n^2/4$ entries of M that lie in non-$\tau/4$-homogeneous blocks of $\mathcal{P}$ and at most $\tau n^2/4$ entries of M that lie in $\tau/4$-homogeneous blocks of $\mathcal{P}$ but do not agree with the value of the block. Therefore, the number of entries as above is no more than $\tau n^2/2$, and so there exists a set of $\tau n^2/2$ pairwise disjoint copies of A in M separated by $X \times Y$ in which all the entries come from $\tau/4$-homogeneous blocks and agree with the value of the block in which they lie. Hence there exist sets of rows $R_1, \ldots, R_s \in \mathcal{R}'$ and sets of columns $C_1, \ldots, C_t \in \mathcal{C}'$ and a collection $\mathcal{A}$ of $\tau n^2/2(r')^{2k}$ pairwise disjoint A-copies separated by $X \times Y$ such that for any $1 \leq i \leq s$, $1 \leq j \leq t$, the block $R_i \times C_j$ is $\tau/4$-homogeneous, has value $A(i,j)$, lies between row separators x_{i-1} and x_i and between column separators y_{j-1} and y_j, and contains the (i,j) entry of any A-copy in $\mathcal{A}$. This implies that $|R_i|, |C_j| \geq \tau n/2(r')^{2k}$ for any $1 \leq i \leq s$ and $1 \leq j \leq t$, So there are $(\tau/2(r')^{2k})^{s+t} n^{s+t}$ $s \times t$ submatrices of M whose (i,j) entry lies in $R_i \times C_j$ for any i, j. Picking such a submatrix S at random, the probability that $S(i,j) \neq A(i,j)$ for a specific couple i, j is at most $\tau/4$; thus S is equal to A with probability at least $1 - st\tau/4 > 1/2$ for small enough τ. Hence the number of A-copies in M is at least$(\tau/2(r')^{2k})^{s+t} n^{s+t}/2$. ◀

Next we give the proof of Theorem 5. For the proof, recall the definition of an unfoldable matrix and a folding of a matrix from Section 3. A family of matrices is *unfoldable* if all matrices in it are unfoldable. The *folding* of a finite family $\mathcal{F}$ of matrices is the set $\tilde{\mathcal{F}} = \{\tilde{A} : A \in \mathcal{F}\}$ of the foldings of the matrices in $\mathcal{F}$. Observe that $\tilde{\mathcal{F}}$ is unfoldable for any family $\mathcal{F}$. Note that if $\mathcal{F}$ is closed under (row) permutations then $\tilde{\mathcal{F}}$ is also closed under (row) permutations.

We start with a short sketch of the proof, before turning to the full proof: As before, we may assume that our matrix M has a row clustering with suitable parameters. We may also assume that the forbidden family is unfoldable. Consider a submatrix Q of M that contains exactly one "representative" row from any large enough row cluster. The crucial idea is that if Q does not contain many A-copies, then M is close to $\mathcal{F}$-freeness. Indeed, one can modify all rows in M to be equal to rows from Q without making many entry modifications, and after this modification, it is possible to eliminate all $\mathcal{F}$-copies in M (without creating new $\mathcal{F}$-copies) by only modifying those columns in M that participate in some $\mathcal{F}$-copy in Q; if Q

does not contain many $\mathcal{F}$-copies then the number of such columns is small. Since the above statement is true for any possible choice of Q, we conclude that if M is ϵ-far from $\mathcal{F}$-freeness then it must contain many A-copies.

Proof of Theorem 5. It is enough to prove the statement of the theorem only for *unfoldable* families that are closed under row permutations. Indeed, suppose that Theorem 5 is true for all unfoldable families that are closed under row permutations. Let $\mathcal{F}$ be a family of binary matrices that is closed under row permutations and let $\tilde{\mathcal{F}}$ be its folding. Then for any $\epsilon > 0$ there exists $\tilde{\delta} > 0$ such that any square binary matrix M which is ϵ-far from $\tilde{\mathcal{F}}$-freeness contains $\tilde{\delta} n^{s'+t'}$ copies of some $s' \times t'$ matrix $B \in \tilde{\mathcal{F}}$, where $\tilde{\delta}^{-1}$ is polynomial in ϵ^{-1}. Thus, provided that M is large enough (i.e. that it is an $n \times n$ matrix where $n \geq n_0$ for a suitable choice of n_0 polynomial in ϵ^{-1}), we can apply Lemma 7 to get that M also contains δn^{s+t} copies of the matrix $A \in \mathcal{F}$ whose folding is B, for a small enough $\delta > 0$ where δ^{-1} is polynomial in ϵ^{-1}.

Therefore, suppose that $\mathcal{F}$ is an unfoldable finite family of binary matrices that is closed under row permutations. Let k be the maximal row or column dimension of a matrix from $\mathcal{F}$. Let $\epsilon > 0$ and apply Lemma 9 with parameters k and $\epsilon/6$. Let M be a large enough $n \times n$ matrix with $n > (k/\epsilon)^{O(k)}$, then either M contains $\delta_2 n^{2k}$ copies of any $k \times k$ matrix, where δ_2^{-1} is polynomial in ϵ^{-1}, or M has an $(\epsilon/6, r)$-clustering of the rows with r polynomial in ϵ^{-1}. In the first case we are done, so suppose that M has an $(\epsilon/6, r)$-clustering $\mathcal{R} = \{R_0, \ldots, R_r\}$ of the rows where R_0 is the error cluster.

Suppose that M is ϵ-far from $\mathcal{F}$-freeness. We say that a cluster $R \neq R_0$ in $\mathcal{R}$ is *large* if it contains at least $\epsilon n/6r$ rows. Note that the total number of entries that do not lie in large clusters is at most $\epsilon n/6 + \epsilon n/6 = \epsilon n/3$. Pick an arbitrary row $r(R)$ from every large cluster $R \in \mathcal{R}$ and denote by Q the submatrix of M created by these rows. Let $\mathcal{A}(Q)$ be a collection of pairwise disjoint copies of matrices from $\mathcal{F}$ in Q that has the maximal possible number of copies. Suppose to the contrary that $|\mathcal{A}| \leq \epsilon n/3k$ and let C be the set of all columns of M that intersect a copy from $\mathcal{A}$, then C contains no more than $\epsilon n/3$ columns. We can modify M to make it $\mathcal{F}$-free as follows: First modify every row that lies in a large cluster $R \in \mathcal{R}$ to be equal to $r(R)$. Then pick some row r of Q and modify all rows that are not contained in large clusters to be equal to r. Finally do the following: As long as C is not empty, pick a column $c \in C$ that has a neighbor (predecessor or successor) not in C and modify c to be equal to its neighbor, and then remove c from C.

It is not hard to see that since $\mathcal{F}$ is unfoldable and closed under row permutations, after these modifications M is $\mathcal{F}$-free. Indeed, after the first and the second steps, all rows of M are equal to rows from Q; the order of the rows does not matter since $\mathcal{F}$ is closed under row permutations. Now each time that we modify a column $c \in C$ in the third step, all copies of matrices from $\mathcal{F}$ that intersect it are destroyed and no new copies are created. By the maximality of $\mathcal{A}$, any copy of a matrix from $\mathcal{F}$ in the original Q intersected some column from C, so we are done. The number of entry modifications needed in the first, second, third step respectively is at most $\epsilon n^2/6$, $\epsilon n^2/3$, $\epsilon n^2/3$ and thus by making only $5\epsilon n^2/6$ modifications of entries of M we can make it $\mathcal{F}$-free, contradicting the fact that M is ϵ-far from $\mathcal{F}$-freeness.

Let Q be any matrix of representatives of the large row clusters as above. Then Q contains a collection $\mathcal{A}$ of $\epsilon n/3k$ pairwise disjoint copies of matrices from $\mathcal{F}$. In particular, there exist a certain $s \times n$ submatrix T of Q and an $s \times t$ matrix $A(Q) \in \mathcal{F}$ such that at least $\epsilon n/3k|\mathcal{F}|r^s$ of the copies in $\mathcal{A}$ are A-copies that lie in T. The following elementary removal lemma implies that T contains many A-copies.

▶ **Observation 12.** *Fix an $s \times t$ matrix A. For any $\epsilon > 0$ there exists $\delta > 0$ such that if an $s \times n$ matrix T contains ϵn pairwise-disjoint A-copies, then the total number of A-copies in T is at least δn^t, with δ^{-1} polynomial in ϵ^{-1}.*

Proof. Let $\epsilon > 0$ and let T be a large enough $s \times n$ matrix containing ϵn pairwise disjoint copies of A. We construct t disjoint subcollections $\mathcal{A}_1, \ldots, \mathcal{A}_t$ of $\mathcal{A}$, each of size $\epsilon n/2t \le \lfloor \epsilon n/t \rfloor$, such that for any $i < j$, all copies in $\mathcal{A}_i$ are i-column-smaller than all copies in $\mathcal{A}_j$. This is done by the following process for $i = 1, \ldots, t$: take $\mathcal{A}_i$ to be the set of the $\epsilon n/2t$ i-smallest copies in $\mathcal{A}$ and delete these copies from $\mathcal{A}$. Now observe that any $s \times t$ submatrix of T that takes its i-th column (for $i = 1, \ldots, t$) as the i-th column of some copy from $\mathcal{A}_i$ is equal to A. There are $(\epsilon n/2t)^t$ such submatrices among all $\binom{n}{t} \le n^t$ $s \times t$ submatrices of T, and so T contains $(\epsilon/2t)^t n^t$ A-copies. ◀

Observation 12 implies that for Q and $A(Q)$ as above, Q contains γn^{s+t} A-copies where γ^{-1} is polynomial in $(\epsilon/3k|\mathcal{F}|r^s)^{-1}$ and so in ϵ^{-1}. Finally we show that M contains δn^{s+t} copies of some $A \in \mathcal{F}$ where δ^{-1} is polynomial in γ^{-1} and so in ϵ^{-1}, finishing the proof of the Theorem. For any large cluster $R \in \mathcal{R}$ let R' be some subcluster that contains exactly $\lfloor \epsilon n/6r \rfloor > 0$ rows. Let $\mathcal{R}' = \{R' : R \in \mathcal{R} \text{ is large}\}$ and note that an α-fraction of the $k \times k$ submatrices S of M have all of their rows in subclusters from $\mathcal{R}'$ with no subcluster containing more than one row of S, where α^{-1} is polynomial in ϵ^{-1}. Let S be a random $k \times k$ submatrix of M. Conditioning on the event that S satisfies the above property, we can assume that S is chosen in the following way: First a random Q is created by picking uniformly at random one representative from every $R' \in \mathcal{R}'$, and then S is taken as a random $k \times k$ submatrix of Q. Let $A = A(Q)$ be defined as above. The probability that S contains a copy of A is at least γ. That is, a random $k \times k$ submatrix S of M contains a copy of a matrix from $\mathcal{F}$ with probability at least $\alpha\gamma$, so there exists an $s \times t$ matrix $A \in \mathcal{F}$ that is contained in a randomly chosen such S with probability at least $\alpha\gamma/|\mathcal{F}|$, so M contains $\alpha\gamma\binom{n}{s}\binom{n}{t}/|\mathcal{F}|k^{2k}$ copies of some $A \in \mathcal{F}$: To see this, observe that we can choose a random $s \times t$ submatrix S' of M by first picking a random $k \times k$ submatrix S and then picking an $s \times t$ random submatrix S' of S. The event that S contains a copy of A has probability at least $\alpha\gamma/|\mathcal{F}|$, and conditioned on this event, S' is equal to A with probability at least k^{-2k}, as the number of $s \times t$ submatrices of S is at most $s^k t^k \le k^{2k}$. The proof is concluded by taking a suitable $\delta = \delta(\epsilon) > 0$ that satisfies $\delta n^{s+t} \le \alpha\gamma\binom{n}{s}\binom{n}{t}/|\mathcal{F}|k^{2k}$ for large enough values of n. Note that indeed δ^{-1} is polynomial in ϵ^{-1}. ◀

5 Multi-dimensional matrices over arbitrary alphabets

As opposed to the polynomial dependence in the above results on binary matrices, Fischer and Rozenberg [15] showed that in analogous results for ternary matrices, as well as binary three-dimensional matrices, the dependence is super-polynomial in general. The proof builds on a construction of Behrend [10]. For the ternary case, it gives the following.

▶ **Theorem 13** ([15]). *There exists a (finite) family $\mathcal{F}$ of 2×2 binary matrices that is closed under permutations and satisfies the following. For any small enough $\epsilon > 0$, there exists an arbitrarily large $n \times n$ ternary matrix M that contains ϵn^2 pairwise-disjoint copies of matrices from $\mathcal{F}$, yet the total number of submatrices from $\mathcal{F}$ in M is no more than $\epsilon^{-c \log \epsilon} n^4$ where $c > 0$ is an absolute constant.*

Theorem 13 implies that an analogue of Theorem 2 with polynomial dependence cannot be obtained when the alphabet is bigger than binary, even when $\mathcal{F}$ is a small finite family that is closed under permutations. In Subsection 5.1 we describe another construction that establishes Theorem 13, which is slightly simpler than the original construction in [15].

In what follows, we focus on the problem of finding a "weak" removal lemma analogous to Theorem 2 for matrices in more than two dimensions over an arbitrary alphabet. Here we do not try to optimize the dependence between the parameters, but rather to show that such a removal lemma exists. Note that in two dimensions this removal lemma follows from Theorem 1, but our results here suggest a direction to prove a weak high dimensional removal lemma without trying to generalize the heavy machinery used in [2] to the high dimensional setting. Our main result here states that this problem is equivalent in some sense to the problem of showing that if a hypermatrix M contains many pairwise-disjoint copies of a hypermatrix A, then it contains a "wide" copy of A; more details are given later. In what follows, we use the term d-matrix to refer to a matrix in d dimensions. An (n, d)-*matrix* is a d-matrix whose dimensions are $n \times \cdots \times n$.

A weak removal lemma for families of d-matrices that are closed under permutations follows easily from the hypergraph removal lemma [19, 22, 21, 27] using a suitable construction.

▶ **Proposition 14.** *Let Γ be an arbitrary alphabet and let $\mathcal{F}$ be a finite family of d-matrices over Γ that is closed under permutations (in all d coordinates). For any $\epsilon > 0$ there exists $\delta > 0$ such that the following holds. If an (n, d)-matrix M over Γ contains ϵn^d pairwise disjoint copies of d-matrices from $\mathcal{F}$, then M contains $\delta n^{s_1+\cdots+s_d}$ copies of some $s_1 \times \ldots \times s_d$ matrix $A \in \mathcal{F}$.*

Note that Theorem 13 implies that the dependence of δ^{-1} on ϵ^{-1} in Proposition 14 cannot be polynomial. The question whether the statement of Proposition 14 holds for any finite family F is open for d-matrices with $d > 2$. Here we state the question in the following equivalent but simpler form.

▶ **Problem 15.** *Let $d > 2$ be an integer. Is it true that for any alphabet Γ, $s_1 \times \ldots \times s_d$ matrix A over Γ and $\epsilon > 0$ there exists $\delta > 0$, such that for any (n, d)-matrix M over Γ containing ϵn^d pairwise-disjoint copies of A, the total number of A-copies in M is at least $\delta n^{s_1+\cdots+s_d}$?*

Note that Theorem 2 settles the two-dimensional binary case of Problem 15 with δ^{-1} polynomial in ϵ^{-1}, and Theorem 1 settles the two-dimensional case over any alphabet. On the other hand, δ^{-1} cannot be polynomial in ϵ^{-1} if $|\Gamma| > 2$ or $d > 2$.

Our main theorem in this domain shows that Problem 15 is equivalent to another statement that looks more accessible. We need the following definition to describe it. Let $M : [n_1] \times \ldots \times [n_d] \to \Gamma$ and let S be the submatrix of M on the indices $\{r^1_1, \ldots, r^1_{s_1}\} \times \ldots \times \{r^d_1, \ldots, r^d_{s_d}\}$ where $r^i_j < r^i_{j+1}$ for any $1 \le i \le d$, $1 \le j \le s_i - 1$. The *(i, j)-width* of S (for $1 \le i \le d$ and $1 \le j \le s_i - 1$) is $(r^i_{j+1} - r^i_j)/n_i$.

▶ **Theorem 16.** *The following statements are equivalent for any $d \ge 2$.*

1. *For any alphabet Γ, $s_1 \times \ldots \times s_d$ matrix A over Γ and $\epsilon > 0$ there exists $\delta > 0$ such that for any (n, d)-matrix M that contains ϵn^d pairwise disjoint copies of A, the total number of A-copies in M is at least $\delta n^{s_1+\cdots+s_d}$.*
2. *For any alphabet Γ, $s_1 \times \ldots \times s_d$ matrix A over Γ and $\epsilon > 0$ there exists $\delta > 0$ such that for any (n, d)-matrix M that contains ϵn^d pairwise disjoint copies of A, and any $1 \le i \le d$, $1 \le j \le s_i$, there exists an A-copy in M whose (i, j)-width is at least δ.*

The proofs of the statements here are given, for simplicity, only for two dimensional matrices, but they translate directly to higher dimensions. The only major difference in the high dimensional case is the use of the hypergraph removal lemma instead of the graph removal lemma. Due to space considerations, the proof of Theorem 16 is relegated to Appendix A, and here we only give the proof of Proposition 14.

Some definitions are required for the proof of Proposition 14. An $s \times t$ *reordering* σ is a permutation of $[s] \times [t]$ that is a Cartesian product of two permutations $\sigma_1 : [s] \to [s]$ and $\sigma_2 : [t] \to [t]$. Given an $s \times t$ matrix A, the $s \times t$ matrix $\sigma(A)$ is the result of the following procedure: First reorder the rows of A according to the permutation σ_1 and then reorder the columns of the resulting matrix according to the permutation σ_2.

Proof of Proposition 14. Let $k(\mathcal{F})$ denote the largest row or column dimension of matrices from $\mathcal{F}$. Let $\epsilon > 0$ and let M be an $n \times n$ matrix over Γ that contains ϵn^2 pairwise-disjoint copies of matrices from $\mathcal{F}$. In particular, there is an $s \times t$ matrix $A \in \mathcal{F}$ such that M contains $\epsilon n^2/|\mathcal{F}|$ pairwise-disjoint copies of A.

We construct an $(s+t)$-partite graph G on $(s+t)n$ vertices as follows: There are s row parts $R_1, \ldots, R_s$ and t column parts $T_1, \ldots, T_t$, each containing n vertices. The vertices of R_i (C_i) are labeled $r^i_1, \ldots, r^i_n$ ($c^i_1, \ldots, c^i_n$ respectively). Two vertices r^a_i and r^b_j (or c^a_i and c^b_j) with $a \neq b$ are connected by an edge iff $i \neq j$. r^a_i and c^b_j are connected iff $M(i,j) = A(a,b)$.

We now show that there exists a bijection between copies of K_{s+t} in G and couples (S, σ) where S is an $s \times t$ submatrix of M and σ is an $s \times t$ reordering such that $\sigma(S) = A$. Indeed, take the following mapping: A couple (S, σ), where S is the submatrix of M on $\{a_1, \ldots, a_s\} \times \{b_1, \ldots, b_t\}$ with $a_1 < \ldots < a_s$ and $b_1 < \ldots < b_t$ and $\sigma = \sigma_1 \times \sigma_2$, is mapped to the induced subgraph of G on $\{r^{\sigma_1(1)}_{a_1}, \ldots, r^{\sigma_1(s)}_{a_s}, c^{\sigma_2(1)}_{b_1}, \ldots, c^{\sigma_2(t)}_{b_t}\}$.

It is not hard to see that (S, σ) is mapped to a copy of K_{s+t} if and only if $\sigma(S)$ is equal to A. On the other hand, every copy of K_{s+t} in G has exactly one vertex in each row part and in each column part, and there exists a unique couple (S, σ) mapped to it.

There exist $\epsilon n^2/|\mathcal{F}|$ pairwise-disjoint A-copies in M that are mapped (with the identity reordering) to edge-disjoint copies of K_{s+t} in G. By the graph removal lemma, there exists $\delta > 0$ such that at least a δ-fraction of the subgraphs of G on $s+t$ vertices are cliques. Therefore, at least a δ-fraction of the possible couples (S, σ) (where S is an $s \times t$ submatrix of M and σ is an $s \times t$ reordering) satisfy $\sigma(S) = A$, concluding the proof. ◀

The proof of Theorem 16 is given in Appendix A.

5.1 Lower bound

In this subsection we give an alternative constructive proof of Theorem 13. Our main tool is the following result in additive number theory from [1], based on a construction of Behrend [10].

▶ **Lemma 17** ([1, 10]). *For every positive integer m there exists a subset $X \subseteq [m] = \{1, \ldots, m\}$ with no non-trivial solution to the equation $x_1 + x_2 + x_3 = 3x_4$, where X is of size at least*

$$|X| \geq \frac{m}{e^{20\sqrt{\log m}}}. \tag{1}$$

Proof of Theorem 13. Consider the family $\mathcal{F} = \{A, B\}$ where

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

and observe that $\mathcal{F}$ is closed under permutations. Let m be a positive integer divisible by 10 and let $X \subseteq [m/10]$ be a subset with no non-trivial solution to the equation $x_1+x_2+x_3 = 3x_4$

that is of maximal size. We construct the following $m \times m$ ternary matrix M. For any $1 \leq i \leq m/5$ and any $x \in X$ we put a copy of A in M as follows:

$$M(i, i+x) = M(m/2+i+2x, m/2+i+3x) = 1$$
$$M(i, m/2+i+3x) = M(m/2+i+2x, i+x) = 0.$$

We set all other entries of M to 2. Let $\mathcal{A}$ be the collection of $q = m|X|/5 \geq m^2/50e^{20\sqrt{\log m}}$ pairwise disjoint copies of A in M that are created as above. Note that all A-copies in M are separated by $\{n/2\} \times \{n/2\}$, where there are two opposite quarters (with respect to the separation) that do not contain the entry 0 and the two other opposite quarters do not contain 1. Hence, every A-copy must contain one entry from each quarter, and M does not contain copies of B. The main observation is that all of the A-copies in M are actually copies from $\mathcal{A}$, so M contains exactly q A-copies.

To see this, suppose that the rows of an A-copy in M are i and $j + n/2$ for some $1 \leq i, j \leq n/2$, then there exist $x_1, x_2, x_3, x_4 \in X$ such that the entries of the copy were taken from locations $(i, i+x_1), (i, m/2+i+3x_2), (m/2+j, j-x_3), (m/2+j, m/2+j+x_4)$ in M and so we have $i + x_1 = j - x_3$ and $i + 3x_2 = j + x_4$. Reordering these two equations we get that $3x_2 = x_1 + x_3 + x_4$, implying that $x_1 = x_2 = x_3 = x_4$ and $j = i + 2x_1$, so the above A-copy is indeed in $\mathcal{A}$.

Let n be an arbitrarily large positive integer divisible by m. Given M as above, we create an $n \times n$ 'blowup' matrix N as follows: For any $1 \leq i, j \leq n$, $N(i,j) = M(\lfloor im/n \rfloor, \lfloor jm/n \rfloor)$. N can also be seen as the result of replacing any entry e in M with an $n/m \times n/m$ matrix of entries equal to e. The total number of A-copies in N is exactly $(n/m)^4 q = n^4|X|/5m^3$, whereas the maximum number of pairwise disjoint A-copies in N is exactly $(n/m)^2 q = n^2|X|/5m$. Assuming that $\epsilon > 0$ is small enough and picking m to be the smallest integer divisible by 10 and larger than $\epsilon^{c \log \epsilon}$ for a suitable absolute constant $c > 0$ gives that $|X|/5m > \epsilon$, but the number of A-copies in N is at most $n^4|X|/5m^3 \leq n^4/m^2 < \epsilon^{-c \log \epsilon} n^4$ as needed. ◀

6 Concluding remarks

Generally, understanding property testing seems to be easier for objects that are highly symmetric. A good example of this phenomenon is the problem of testing properties of (ordered) one-dimensional binary vectors. There are some results on this subject, but it is far from being well understood. On the other hand, the binary vector properties P that are invariant under permutations of the entries (these are the properties in which for any vector v that satisfies P, any permutation of the entries of v also satisfies P) are merely those that depend only on the length and the Hamming weight of a vector. This makes the task of testing these properties trivial.

A central example of the symmetry phenomenon is the well investigated subject of property testing in (unordered) graphs, that considers only properties of functions from $\binom{[n]}{2}$ to $\{0,1\}$ that are invariant under permutations of $\binom{[n]}{2}$ induced by permutations on $[n]$. That is, if a labeled graph G satisfies some graph property, then any relabeling of its vertices results in a graph that also satisfies this property. Indeed, the proof of the only known general result on testing properties of *ordered graphs* (here the functions are generally not invariant under permutations), given in [2], is substantially more complicated than the proof of its unordered analogue. See [25] for further discussion on the role of symmetries in property testing.

In general, matrices (with row and column order) do not have any symmetries. Therefore, the above reasoning suggests that proving results on the testability of matrix properties is likely to be harder than proving similar results on properties of matrices where only the rows are ordered (such properties are invariant under permutations of the columns), which might be harder in turn than proving the same results for properties of matrices without row and column orders, i.e. bipartite graphs, as these properties are invariant under permutations of both the rows and the columns.

Theorem 2 is a weak removal lemma for binary matrices with row and column order, while Theorem 3 is an induced removal lemma for binary matrices without row and column order, and our generalization of it, Theorem 5, is an induced removal lemma for binary matrices with a row order but without a column order. It will be very interesting to settle Problem 4, that asks whether a polynomial induced removal lemma exists for binary matrices with row and column orders.

It will be interesting to expand our knowledge of matrices in higher dimensions and of ordered combinatorial objects in general. Proposition 14 is a non-induced removal lemma for (multi-dimensional) matrices without row and column orders. It will be interesting to get results of this type for less symmetric objects, ultimately for ordered multi-dimensional matrices. We believe that providing a direct solution (that does not go through Theorem 1) for the following seemingly innocent problem is of interest, and might help providing techniques to help settling Problem 15 in general. In what follows, the *height* of a 2×2 submatrix S in an $n \times n$ matrix M is the difference between the indices of the rows of S in M, divided by n.

▶ **Problem 18.** *Let* $A = \begin{pmatrix} 0 & 1 \\ 2 & 3 \end{pmatrix}$ *and suppose that an* $n \times n$ *matrix contains* ϵn^2 *pairwise disjoint copies of* A*. Show (without relying on Theorem 1) that there exists* $\delta = \delta(\epsilon)$ *such that* M *contains an* A*-copy with height at least* δ*.*

The three dimensional analogue of this problem is obviously also of interest. Here Theorem 1 cannot be applied, so currently we do not know whether such a $\delta = \delta(\epsilon)$ that depends only on ϵ exists. Solving the three-dimensional analogue will settle Problem 15 when the forbidden hypermatrix has dimensions $2 \times 2 \times 2$, and the techniques might lead to settling Problem 15 in its most general form.

As a final remark, in the results in which δ^{-1} is polynomial in ϵ^{-1} we have not tried to obtain tight bounds on the dependence, and it may be interesting to do so.

Acknowledgements. The authors wish to thank the anonymous reviewers for useful feedback.

References

1 N. Alon. Testing subgraphs in large graphs. *Random Structures and Algorithms*, 21:359–370, 2002.

2 N. Alon, O. Ben-Eliezer, and E. Fischer. Testing hereditary properties of ordered graphs and matrices. *arXiv*, 1704:02367, 2017.

3 N. Alon, R. A. Duke, H. Lefmann, V. Rödl, and R. Yuster. The algorithmic aspects of the regularity lemma. *Journal of Algorithms*, 16:80–109, 1994.

4 N. Alon, E. Fischer, M. Krivelevich, and M. Szegedy. Efficient testing of large graphs. *Combinatorica*, 20:451–476, 2000.

5 N. Alon, E. Fischer, and I. Newman. Efficient testing of bipartite graphs for forbidden induced subgraphs. *SIAM J. Comput.*, 37:959–976, 2007.

6 N. Alon and J. Fox. Easily testable graph properties. *Combin. Probab. Comput*, 24:646–657, 2015.

7 N. Alon, M. Krivelevich, I. Newman, and M. Szegedy. Regular languages are testable with a constant number of queries. *SIAM J. Comput.*, 30:1842–1862, 2001.

8 N. Alon and A. Shapira. A characterization of easily testable induced subgraphs. *Combin. Probab. Comput.*, 15:791–805, 2006.

9 N. Alon and A. Shapira. A characterization of easily testable induced subgraphs. *SIAM J. Comput.*, 37:1703–1727, 2008.

10 F. Behrend. On sets of integers which contain no three terms in arithmetic progression. *Proc. Nat. Acad. Sci.*, 32:331–332, 1946.

11 O. Ben-Eliezer, S. Korman, and D. Reichman. Deleting and Testing Forbidden Patterns in Multi-Dimensional Arrays. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*, volume 80 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 9:1–9:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2017.9.

12 D. Conlon and J. Fox. Bounds for graph regularity and removal lemmas. *Geom. Funct. Anal.*, 22:1191–1256, 2012.

13 D. Conlon and J. Fox. Graph removal lemmas. In *Surveys in Combinatorics*, pages 1–49. Cambridge Univ. Press, 2013.

14 E. Fischer and I. Newman. Testing of matrix-poset properties. *Combinatorica*, 27:293–327, 2007.

15 E. Fischer and E. Rozenberg. Lower bounds for testing forbidden induced substructures in bipartite-graph-like combinatorial objects. In *Proc. 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Princeton, NJ, USA, August 20-22, 2007*, pages 464–478. Springer Berlin, Heidelberg, 2007.

16 J. Fox. A new proof of the graph removal lemma. *Ann. of Math.*, 174:561–579, 2011.

17 L. Gishboliner and A. Shapira. Removal lemmas with polynomial bounds. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 510–522. ACM, 2017. doi:10.1145/3055399.3055404.

18 O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45:653–750, 1998.

19 T. Gowers. A new proof of Szemerédi's theorem. *Geom. Funct. Anal.*, 11:465–588, 2001.

20 L. Lovász and B. Szegedy. Regularity partitions and the topology of graphons. In *An irregular mind*, pages 415–445. János Bolyai Math. Soc., Budapest, 2010.

21 B. Nagle, V. Rődl, and M. Schacht. The counting lemma for regular k-uniform hypergraphs. *Random Structures and Algorithms*, 28:113–179, 2006.

22 V. Rödl and J. Skokan. Regularity lemma for uniform hypergraphs. *Random Structures and Algorithms*, 25:1–42, 2004.

23 R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *Siam J. Comput.*, 25:252–271, 1996.

24 I. Z. Ruzsa and E. Szemerédi. Triple systems with no six points carrying three triangles. In *Combinatorics, Vol. II*, pages 939–945. Coll. Math. Soc. J. Bolyai 18, North-Holland, Amsterdam-New York, 1978.

25 M. Sudan. Invariance in property testing. In *Property Testing: Current Research and Surveys*, pages 211–227. Springer Brelin, Heidelberg, 2010.

26 E. Szemerédi. Regular partitions of graphs. In *Problèmes Combinatoires et Théorie des Graphes*, pages 399–401. Colloq. Internat. CNRS 260, Orsay, 1976.

27 T. Tao. A variant of the hypergraph removal lemma. *J. Combin. Theory Ser. A*, 113:1257–1280, 2006.

A Proof of Theorem 16

The proof is given for the two-dimensional case, but it translates to the high-dimensional case naturally (using the hypergraph removal lemma instead of the graph removal lemma). We may and will assume throughout the proof that M is an $n \times n$ matrix where n is large enough with respect to ϵ. The terms i-height and j-width correspond to $(1, i)$-width and $(2, j)$-width, respectively, in the definition given before the statement of Theorem 16.

Proof of Theorem 16. We start with deriving Statement 2 from Statement 1; this direction is quite straightforward, while the other direction is more interesting. Fix an $s \times t$ matrix A. Let $\epsilon > 0$ and assume that Statement 1 holds. There exists $\delta = \delta(\epsilon)$, such that if M contains ϵn^2 pairwise-disjoint A-copies then it contains δn^{s+t} copies of A. To prove Statement 2 we can pick $\delta' = \delta'(\epsilon) > 0$ small enough such that for any large enough $n \times n$ matrix M, any $1 \le i \le s-1$ and any $1 \le j \le t-1$, the fraction of $s \times t$ submatrices with i-height (or j-width) smaller than δ' among all $s \times t$ submatrices is at most $\delta/2$. Fix an $1 \le i \le s-1$. This choice of δ' implies that any matrix M containing ϵn^2 pairwise disjoint A-copies also contains an A-copy with i-height at least δ'. Similarly, for any $1 \le j \le t-1$ there is an A-copy with j-width at least δ'.

Next we assume that Statement 2 holds and prove Statement 1. Fix an $s \times t$ matrix A over an alphabet Γ, let $\epsilon > 0$ and let M be a large enough $n \times n$ matrix containing a collection $\mathcal{A}_0$ of ϵn^2 pairwise disjoint A-copies. We will show that there exist $\epsilon^* > 0$ that depends only on ϵ, sets X, Y of row and column separators respectively of sizes $s-1$ and $t-1$ and a collection of $\epsilon^* n^2$ disjoint A-copies separated by $X \times Y$ in M. Then we will combine a simpler variant of the construction used in the proof of Proposition 14 with the graph removal lemma to show that M contains δn^{s+t} copies of A for a suitable $\delta(\epsilon) > 0$.

The number of A-copies in M does not depend on the alphabet, so we may consider A and M as matrices over the alphabet $\Gamma' = \Gamma \cup \{\alpha\}$ for some $\alpha \notin \Gamma$, even though all symbols in A and M are from Γ. Without loss of generality we assume that no two entries in A are equal.

Let $X_0 = \phi$, $\epsilon_0 = \epsilon$ and let M_0 be the following $n \times n$ matrix over Γ': All A-copies in $\mathcal{A}_0$ appear in the same locations in M_0, and all other entries of M_0 are equal to α. Clearly, any A-copy in M_0 also appears in M. Next, we construct iteratively for any $i = 1, \dots, s-1$ an $n \times n$ matrix M_i over Γ' that contains a collection $\mathcal{A}_i$ of $\epsilon_i n^2$ pairwise disjoint copies of A where $\epsilon_i > 0$ depends only on ϵ_{i-1}, such that all A-copies in M_i also exist in M_{i-1}. We also maintain a set X_i of row separators whose elements are $x_1 < \ldots < x_i$, such that any entry of M_i between x_{j-1} and x_j for $j = 1, \dots, i$ (where we define $x_0 = 0, x_s = n$) is either equal to one of the entries of the j-th row of A or to α.

The construction of M_i given M_{i-1} is done as follows. By Statement 2, there exists $\delta_i = \delta_i(\epsilon_{i-1})$ such that any matrix M' over Γ' containing at least $\epsilon_{i-1} n^2/2$ copies of A also contains a copy of A with i-height at least δ_i. We start with a matrix M' equal to M_{i-1} and an empty $\mathcal{A}_i$, and as long as M' contains a copy of A with i-height at least δ_i, we add it to $\mathcal{A}_i$ and modify (in M') all entries of all A-copies from $\mathcal{A}_{i-1}$ that intersect it to α. By the separation that X_{i-1} induces on M', each such copy has its j-th row between x_{j-1} and x_j for any $1 \le j \le i-1$.

This process might stop only when at least $\epsilon_{i-1} n^2/2$ of the copies from $\mathcal{A}_{i-1}$ in M' have one of their entries modified. Since in each step at most st copies of A are deleted from M', in the end $\mathcal{A}_i$ contains at least $\epsilon_{i-1} n^2/2st$ pairwise disjoint copies of A with i-height at least δ_i. Pick uniformly at random a row index $x_i > x_{i-1}$. The probability that a certain copy of A in $\mathcal{A}_i$ has its i-th row at or above x_i and its $(i+1)$-th row below x_i is at least δ_i.

Therefore, the expected number of A-copies in $\mathcal{A}_i$ with this property is at least $\epsilon_i n^2$ with $\epsilon_i = \delta_i \epsilon_{i-1}/2st$, so there exists some x_i such that at least $\epsilon_i n^2$ A-copies in $\mathcal{A}_i$ have their first $i+1$ rows separated by $X_i = X_{i-1} \cup \{x_i\}$; delete all other copies from $\mathcal{A}_i$. We construct M_i as follows: All A-copies from $\mathcal{A}_i$ appear in the same locations in M_i, and all other entries of M_i are equal to α.

After iteration $s-1$ we have a matrix M_{s-1} with $\epsilon_{s-1}n^2$ copies of A separated by $X = X_{s-1}$. We apply the same process in columns instead of rows, starting with the matrix M_{s-1}. The resulting matrix M^* contains $\epsilon^* n^2$ pairwise disjoint copies of A separated by $X \times Y$ where Y consists of the column separators $y_1 < \ldots < y_{t-1}$, ϵ^* depends on ϵ, and M^* only contains A-copies that appeared in the original M.

Finally, construct an $(s+t)$-partite graph G on $2n$ vertices as follows: The row parts are $R_1, \ldots, R_s$ and the column parts are $C_1, \ldots, C_t$ where R_i (C_i) contains vertices labeled $x_{i-1}+1, \ldots, x_i$ $(y_{i-1}+1, \ldots, y_i$ respectively) with $x_0 = y_0 = 0, x_s = y_t = n$. Any two row (column) vertices not in the same part are connected. Vertices $a \in R_i, b \in C_j$ are connected if and only if $M^*(a,b) = A(i,j)$. Clearly there exists a bijection between A-copies in M^* and K_{s+t} copies in G that maps disjoint A-copies to edge disjoint K_{s+t}-copies in G, so it contains $\epsilon^* n^2$ edge disjoint $(s+t)$-cliques. By the graph removal lemma there exists $\delta = \delta(\epsilon^*) > 0$ such that a δ-fraction of the subgraphs of G on $s+t$ vertices are cliques. Hence at least a δ-fraction of the $s \times t$ submatrices of M are equal to A. ◀

The String of Diamonds Is Tight for Rumor Spreading*

Omer Angel[1], Abbas Mehrabian[2], and Yuval Peres[3]

1 **Department of Mathematics, University of British Columbia, Vancouver, BC, Canada**
angel@math.ubc.ca

2 **Department of Computer Science, University of British Columbia, Vancouver, BC, Canada**
abbasmehrabian@gmail.com

3 **Microsoft Research, Redmond, WA, USA**
peres@microsoft.com

Abstract

For a rumor spreading protocol, the spread time is defined as the first time that everyone learns the rumor. We compare the synchronous push&pull rumor spreading protocol with its asynchronous variant, and show that for any n-vertex graph and any starting vertex, the ratio between their expected spread times is bounded by $O\left(n^{1/3}\log^{2/3} n\right)$. This improves the $O(\sqrt{n})$ upper bound of Giakkoupis, Nazari, and Woelfel (in Proceedings of ACM Symposium on Principles of Distributed Computing, 2016). Our bound is tight up to a factor of $O(\log n)$, as illustrated by the string of diamonds graph.

1998 ACM Subject Classification C.2.1 Network Architecture and Design, G.2.2 Graph Theory, G.3 Probability and Statistics

Keywords and phrases randomized rumor spreading, push&pull protocol, asynchronous time model, string of diamonds

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.26

1 Introduction

Randomized rumor spreading is an important paradigm for information dissemination in networks with numerous applications in network science, ranging from spreading information in the WWW and Twitter to spreading viruses and diffusion of ideas in human communities. A well studied rumor spreading protocol is the *(synchronous) push&pull protocol*, introduced by Demers, Greene, Hauser, Irish, Larson, Shenker, Sturgis, Swinehart, and Terry [4] and popularized by Karp, Schindelhauer, Shenker, and Vöcking [11].

▶ **Definition 1** (Synchronous push&pull protocol)**.** Suppose that one node s in a network G is aware of a piece of information, the 'rumor', and wants to spread it to all nodes

* This work started in the *Random Geometric Graphs and Their Applications to Complex Networks* workshop held in the Banff International Research Station in November 2016. The authors thank the workshop organizers and BIRS for making it happen. The first author is supported by NSERC. The second author was supported by an NSERC Postdoctoral Fellowship and a Simons-Berkeley Research Fellowship. Part of this work was done while he was visiting the Simons Institute for the Theory of Computing at UC Berkeley. This work was completed at Microsoft Research in Redmond. The authors thank Microsoft for their support.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 26; pp. 26:1–26:9
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

quickly. The protocol proceeds in rounds; in each round $1, 2, \ldots$, all vertices perform actions simultaneously. That is, each vertex x calls a random neighbor y, and the two share any information they may have: If x knows the rumor and y does not, then x tells y the rumor (a *push* operation), and if x does not know the rumor and y knows it, y tells x the rumor (a *pull* operation). Note that this is a synchronous protocol, e.g. a vertex that receives a rumor in a certain round cannot send it on in the same round. The synchronous spread time of G, denoted by $S(G, s)$, is the first time that everyone knows the rumor. Note that this is a discrete random variable.

A point to point communication network can be modeled as an undirected graph: the nodes represent the processors and the links represent communication channels between them. Studying rumor spreading has several applications to distributed computing in such networks, of which we mention just two (see [7] also). The first is in broadcasting algorithms: a single processor wants to broadcast a piece of information to all other processors in the network. The push&pull protocol has several advantages over other protocols: it puts less load on the edges than the naive flooding protocol; it is simple and naturally distributed (each node makes a simple local decision in each round; no knowledge of the global state or topology is needed; no internal states are maintained); it is scalable (the protocol is independent of the size of network: it does not grow more complex as the network grows) and it is robust (the protocol tolerates random node/link failures without the need for error recovery mechanisms).

A second application comes from the maintenance of databases replicated at many sites, e.g., yellow pages, name servers, or server directories. Updates to the database may be injected at various nodes, and these updates must propagate to all nodes in the network. In each round, a processor communicates with a random neighbor and they share any new information, so that eventually all copies of the database converge to the same contents. See [4] for details.

The above protocol assumed a synchronized model, i.e. all nodes take action simultaneously at discrete time steps. In many applications and certainly for modeling information diffusion in social networks, this assumption is not realistic. Boyd, Ghosh, Prabhakar, Shah [3] proposed an asynchronous time model with a continuous time line. This too is a randomized distributed algorithm for spreading a rumor in a graph, defined as follows. An *exponential clock* with rate λ is a clock that, once turned on, rings at times of a Poisson process with rate λ.

▶ **Definition 2** (Asynchronous push&pull protocol)**.** Given a graph G, independent exponential clocks of rate 1 are associated with the vertices of G, one to each vertex. Initially, one vertex s of G knows the rumor, and we turn on all clocks. Whenever the clock of a vertex x rings, it calls a random neighbor y: if x knows the rumor and y does not, then x tells y the rumor (a push operation); if x does not know the rumor and y knows it, y tells x the rumor (a pull operation). The asynchronous spread time of G, denoted by $A(G, s)$, is the first time that everyone knows the rumor.

Rumor spreading protocols in this model turn out to be closely related to Richardson's model for the spread of a disease [12, 6]. For a single rumor, the push&pull protocol is almost equivalent to the first passage percolation model introduced by Hammersley and Welsh [9] with edges having independent exponential weights (see also the survey [2]). The difference between the push&pull model and first passage percolation stems from the fact that in the rumor spreading models each vertex contacts one neighbor at a time, and so the rate at which x pushes the rumor to y is inversely proportional to the degree of x. A rumor

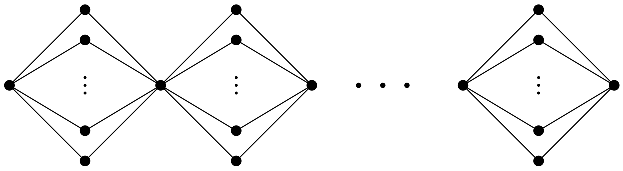


Figure 1 The string of diamonds graph.

can also be pulled from x to y at rate determined by the degree of y. On regular graphs, the asynchronous push&pull protocol, Richardson's model, and first passage percolation are essentially the same process, assuming appropriate parameters are chosen. For general graphs, the equivalence is to first passage percolation with exponential edge weights that are independent, but have different means. Hence, the degrees of vertices play a different role here than they do in Richardson's model or first passage percolation. A collection of known bounds for the average spread times of many graph classes is given in [1, Table 1].

Doerr, Fouz, and Friedrich [5] experimentally compared the spread time in the two time models. They state that 'Our experiments show that the asynchronous model is faster on all graph classes [considered here].' The first general relationship between the spread times of the two variants was given in [1], where it was proved using a coupling argument that

$$\frac{\mathbb{E}\left[S(G,s)\right]}{\mathbb{E}\left[A(G,s)\right]} = \widetilde{O}\left(n^{2/3}\right).$$

Here $\widetilde{O}$ (and $\widetilde{\Omega}$ below) allow for poly-logarithmic factors. Building on the ideas of [1] and using more involved couplings, Giakkoupis, Nazari and Woelfel [8] improved this bound to $O\left(n^{1/2}\right)$. In this note we improve the bound to $\widetilde{O}(n^{1/3})$. A graph was given in [1] with

$$\frac{\mathbb{E}\left[S(G,s)\right]}{\mathbb{E}\left[A(G,s)\right]} = \widetilde{\Omega}\left(n^{1/3}\right),$$

known as the *string of diamonds* (see Figure 1), which shows the exponent 1/3 is optimal.

We use a rather different coupling than previous ones. Our coupling is motivated by viewing rumor spreading as a special case of first passage percolation. This novel approach involves carefully intertwined Poisson processes. Our proof also yields a natural interpretation for the exponent 1/3: using non-trivial counting arguments, we prove that the longest distance the rumor can traverse during a unit time interval in the asynchronous protocol is $O(n^{1/3})$ (see the proof of Lemma 6), which is tight.

Regarding lower bounds, it is proved in [8] that $\mathbb{E}\left[A(G,s)\right] \le \mathbb{E}\left[S(G,s)\right] + O(\log n)$. We will use the following bounds that hold for all G and s (see [1, Theorem 1.3]):

$$\log n/5 \le \mathbb{E}\left[A(G,s)\right] \le 4n\,.$$

In this paper n always denotes the number of vertices of the graph, and all logarithms are in natural base.

1.1 Our results

For an n-vertex graph G and a starting vertex s, recall $A(G,s)$ and $S(G,s)$ denote the asynchronous and synchronous spread times, respectively. Our main theorem is the following.

▶ **Theorem 3.** *Given any $K > 0$, there is a $C > 0$ such that for any (G, s) and any $t \geq 1$ we have*

$$\mathbb{P}\Big[S(G,s) > C(t + t^{2/3}n^{1/3}\log n)\Big] \leq \mathbb{P}\Big[A(G,s) > t\Big] + Cn^{-K}.$$

▶ **Corollary 4.** *For any (G, s), we have $\mathbb{E}\left[S(G,s)\right] = O\left(\mathbb{E}\left[A(G,s)\right]^{2/3} n^{1/3}\log n\right)$.*

Proof. Apply Theorem 3 with $K = 1$ and $t = 3\mathbb{E}\left[A(G,s)\right] \leq 12n$. By Markov's inequality, $\mathbb{P}\left[S(G,s) > C(t + t^{2/3}n^{1/3}\log n)\right] \leq 1/3 + C/n \leq 1/2$ for n large enough. Since $t = O(n)$, this implies the median of $S(G,s)$, denoted by M, is $O(t^{2/3}n^{1/3}\log n)$. To complete the proof we need only show that $\mathbb{E}\left[S(G,s)\right] = O(M)$.

Consider the protocol which is the same as synchronous push&pull except that, if the rumor has not spread to all vertices by time M, then the new process reinitializes. Coupling the new process with push&pull, we obtain for any $i \in \{0, 1, 2, \dots\}$ that $\mathbb{P}\left[S(G,s) > iM\right] \leq 2^{-i}$. Thus,

$$\mathbb{E}\left[S(G,s)\right] = \sum_{i=0}^{\infty}\mathbb{P}\left[S(G,s) > i\right] \leq \sum_{i=0}^{\infty} M \times \mathbb{P}\left[S(G,s) > iM\right] \leq M \times \sum_{i=0}^{\infty} 2^{-i} = 2M. \quad ◀$$

Since for all G and s, $\mathbb{E}\left[A(G,s)\right] = \Omega(\log n)$, we also obtain:

▶ **Corollary 5.** *For any (G, s) we have*

$$\frac{\mathbb{E}\left[S(G,s)\right]}{\mathbb{E}\left[A(G,s)\right]} = O\left(n^{1/3}\log^{2/3} n\right).$$

This corollary is tight up to logarithmic factors. Indeed, let G be the string of diamonds (see Figure 1) with m diamonds, each consisting of k paths of length 2, and let s be an end vertex of it. Then, $S(G,s) \geq 2m$ deterministically and $\mathbb{E}\left[A(G,s)\right] = O(\log n + m/\sqrt{k})$ (see [1] for the proof). If we let $m = \Theta(n^{1/3}(\log n)^{2/3})$ and $k = \Theta((n/\log n)^{2/3})$, we obtain a graph with

$$\frac{\mathbb{E}\left[S(G,s)\right]}{\mathbb{E}\left[A(G,s)\right]} = \Omega\left(n/\log n\right)^{1/3},$$

which means Corollary 5 is tight up to an $O(\log n)$ factor.

2 Proof of Theorem 3

For the rest of the paper we fix the graph G and the starting vertex s. Let $\Gamma(s,v)$ be the set of all simple paths in G from s to v. For a path γ, let $E(\gamma)$ be its set of edges and $|\gamma| := |E(\gamma)|$ denote its length. Let $\deg(u)$ denote the degree of a vertex u.

For any ordered pair (u,v) of adjacent vertices, let $Y_{u,v}$ be an exponential random variable with rate $1/\deg(u)$. Assume these random variables are mutually independent. In the asynchronous protocol, since each vertex u calls any adjacent v at a rate of $1/\deg(u)$, we can write:

$$A := A(G,s) = \max_{v\in V}\min_{\gamma\in\Gamma(s,v)}\sum_{xy\in E(\Gamma)}\min\{Y_{x,y}, Y_{y,x}\}. \tag{1}$$

Here $Y_{x,y}$ is the time it takes after one of x, y learns the rumor before x calls y.

For any positive integer L, consider the restriction to short paths

$$A_L := \max_{v\in V} \min_{\substack{\gamma\in\Gamma(s,v)\\|\gamma|\le L}} \sum_{xy\in E(\Gamma)} \min\{Y_{x,y}, Y_{y,x}\}.$$

For any L we have $A_L \geq A$. To bound A from below, we have the following "with high probability" stochastic domination result.

▶ **Lemma 6.** *There exists a $C > 0$ such that for any $t \geq 1$ and $L \geq Ct^{2/3}n^{1/3}$ we have*

$$\mathbb{P}[A_L > t] \leq \mathbb{P}[A > t] + e^{-L}.$$

Proof. We show that, in the asynchronous protocol, with probability $1-e^{-L}$, during the interval $[0,t]$, the rumor does not travel along any simple path of length L. We prove this by taking a union bound over all paths of length L. As there is no simple path of length n or more, we will assume $L < n$.

Consider a path γ with vertices $\gamma_0, \gamma_1, \ldots, \gamma_L$. In order for the rumor to travel along γ, it is necessary that there are calls along the edges of γ in order, at some sequence of times $0 \leq t_1 < \cdots < t_L \leq t$. Since along each edge the rumor can travel via a push or a pull, the rate of calls along an edge xy is $1/\deg(x) + 1/\deg(y)$. Since the volume of the L-dimensional simplex of possible sequences (t_i) is $t^L/L!$, the probability of such a sequence of calls along the path γ is at most

$$\frac{t^L}{L!}\prod_{i=1}^{L}\left(\frac{1}{\deg(\gamma_{i-1})}+\frac{1}{\deg(\gamma_i)}\right) \leq \left(\frac{2et}{L}\right)^L \prod_{i=1}^{L}\frac{1}{\min(\deg(\gamma_{i-1}),\deg(\gamma_i))}. \tag{2}$$

In light of this, define

$$Q(\gamma) := \prod_{i=1}^{|\gamma|}\frac{1}{\min(\deg(\gamma_{i-1}),\deg(\gamma_i))}.$$

Our objective is therefore a bound for $\sum_{|\gamma|=L} Q(\gamma)$.

For a path γ of length L, consider the sequence of degrees $(\deg(\gamma_i))_{i=0}^{L}$. We say the sequence has a *local minimum* at i if $\deg(\gamma_{i-1}) > \deg(\gamma_i) \leq \deg(\gamma_{i+1})$, and a *local maximum* at i if $\deg(\gamma_{i-1}) \leq \deg(\gamma_i) > \deg(\gamma_{i+1})$. In both of these definitions we use the convention that inequalities involving γ_{-1} or γ_{L+1} always hold. The edge set of γ can be partitioned into *segments* starting and ending at local maxima. For example, suppose $L = 7$ and the degree sequence is

$$(\deg(\gamma_0),\deg(\gamma_1),\deg(\gamma_2),\deg(\gamma_3),\deg(\gamma_4),\deg(\gamma_5),\deg(\gamma_6),\deg(\gamma_7)) = (\mathbf{5},5,7,\mathbf{3},4,4,\mathbf{2},5).$$

Then the segments are $(\boldsymbol{\gamma_0},\gamma_1,\gamma_2)$, $(\gamma_2,\boldsymbol{\gamma_3},\gamma_4,\gamma_5)$, and $(\gamma_5,\boldsymbol{\gamma_6},\gamma_7)$. Thus, in each segment the degrees strictly decrease to a local minimum (bolded in the example), then weakly increase up to the local maximum at the end of the segment. (The first and last segments are special in that the local minimum could be at the beginning and end of the segment, respectively.) Henceforth, we use the term *segment* for a path with this property.

Each path gives rise to an ordered sequence of segments. Denote the segments of γ by $\sigma_1, \ldots, \sigma_s$, and note that $s \leq L/2 + 1$, since each segment except possibly the first and the last one contains at least two edges. The next observation is that we have $Q(\gamma) = \prod Q(\sigma_i)$; that is, the Q value of a path equals the product of Q values of its segments (this is true for any partition of a path into sub-paths). Note that not every sequence of segments can arise in this way: each segment must start at the last vertex of the previous segment. Since we are interested only in simple paths, the segments are otherwise disjoint. Thus for a collection of segments there is at most one order in which it could arise. Therefore,

$$\sum_{|\gamma|=L} Q(\gamma) \le \sum_{s=1}^{L/2+1} \sum_{|\sigma_1|+\cdots+|\sigma_s|=L} \frac{1}{s!} \prod_{i=1}^{s} Q(\sigma_i)\,, \tag{3}$$

where the last sum is over s-tuples of segments whose lengths add up to L, but *without* the condition that they form a path (that is why we have an inequality rather than an equality).

We now bound the right-hand-side of (3). We say a segment has *type* $(x, \ell^-, \ell^+) \in V(G) \times \mathbb{Z} \times \mathbb{Z}$ if the local minimum is at a vertex x (called the *center* of the segment), and the segment has ℓ^- edges before x and ℓ^+ edges after x. (The example path above had $s = 3$ segments, of types $(\gamma_0, 0, 2)$, $(\gamma_3, 1, 2)$, and $(\gamma_6, 1, 1)$ respectively.) For a segment σ, let $\pi(\sigma)$ denote its type, and let $\mathcal{T}$ denote the set of all possible types.

For bounding the right-hand-side of (3), we first fix s and bound the number of options for the sequence $(\pi(\sigma_1), \ldots, \pi(\sigma_s))$. There are $n!/(n-s)!$ choices for the centers (the number of ways to choose s ordered distinct vertices), and at most 2^L choices for the lengths $\ell^\pm$ (the number of ways to write L as an ordered sum of natural numbers). Thus there are at most $2^L n!/(n-s)!$ options for $(\pi(\sigma_1), \ldots, \pi(\sigma_s))$. Enumerate these s-vectors of types by $T_1, \ldots, T_m \in \mathcal{T}^s$ with $m \le 2^L n!/(n-s)!$, and let $T_{j,k}$ denote the kth component of T_j, i.e. the type specified for σ_k in T_j. Thus,

$$\begin{aligned}\sum_{|\sigma_1|+\cdots+|\sigma_s|=L} \prod_{i=1}^{s} Q(\sigma_i) &= \sum_{j=1}^{m} \sum_{(\pi(\sigma_1),\ldots,\pi(\sigma_s))=T_j} \prod_{i=1}^{s} Q(\sigma_i)\\ &\le \sum_{j=1}^{m} \prod_{k=1}^{s} \left(\sum_{\pi(\sigma_k)=T_{j,k}} Q(\sigma_k) \right)\end{aligned}$$

Next, we claim that each of the brackets, which is the sum of Q values of segments of a given type can be bounded by 1. Fix some type (x, ℓ^-, ℓ^+), and let $\ell = \ell^- + \ell^+$. Then the constraints on the degrees along a segment $\sigma = v_0, v_1, \cdots, v_{\ell^-}, \cdots, v_\ell$ of this type imply $x = v_{\ell^-}$ and

$$Q(\sigma) = \prod_{i=1}^{\ell^-} \frac{1}{\deg(v_i)} \prod_{i=\ell^-}^{\ell-1} \frac{1}{\deg(v_i)}.$$

If we sum this up over all *walks* of length $\ell^- + \ell^+$ whose ℓ^-th vertex is x, we get 1 (since the number of choices for the neighbors cancel out the degree reciprocals). Restricting to simple paths with piecewise monotone degrees only decreases this. Thus we obtain

$$\sum_{|\sigma_1|+\cdots+|\sigma_s|=L} \prod_{i=1}^{s} Q(\sigma_i) \le m \times 1 \le 2^L n!/(n-s)!\,.$$

Plugging this back into (3) yields

$$\sum_{|\gamma|=L} Q(\gamma) \le \sum_{s=1}^{L/2+1} 2^L n!/(n-s)!s! = 2^L \sum_{s=1}^{L/2+1} \binom{n}{s} \le 2^L (2en/L)^{L/2+1},$$

where we used the standard bound $\sum_{i=0}^{k} \binom{n}{i} \le (en/k)^k$ valid for all $0 \le k \le n$.

Therefore, by (2), the probability that the rumor travels along some path of length L is bounded by

$$\sum_{|\gamma|=L} \left(\frac{2et}{L}\right)^L Q(\gamma) \le \left(\frac{2et}{L}\right)^L 2^L(2en/L)^{L/2+1} \le c_1 n(c_2 nt^2/L^3)^{L/2},$$

which is at most e^{-L} for $L \ge Ct^{2/3}n^{1/3}$, completing the proof. ◀

In (1) we wrote $A(G,s)$ in a max-min form. We would like to write $S(G,s)$ in a similar way. To achieve this, let $q_{uv} = q_{vu}$ be the first (discrete) round at which one of u or v is informed. Suppose the first round *strictly after* q_{uv} that u calls v is F_{uv}. Then define $T_{u,v} = F_{uv} - q_{uv}$. Note that $T_{u,v}$ is a positive integer, and it is a geometric random variable: $\mathbb{P}[T_{u,v} \ge k] = (1 - 1/\deg(u))^{k-1}$ for any $k = 1, 2, \ldots$. Moreover, observe that, both u and v are informed by round $q_{uv} + \min\{T_{u,v}, T_{v,u}\}$ hence, we have

$$S := S(G,s) \le \max_{v\in V} \min_{\gamma\in\Gamma(s,v)} \sum_{xy\in E(\Gamma)} \min\{T_{x,y}, T_{y,x}\}. \tag{4}$$

Now we have a max-min expression for $S(G,s)$. However, the major trouble here is that the $\{T_{x,y}\}$ are not independent. We will stochastically dominate them by another collection $\{X_{x,y}\}$ of random variables, which are independent. To prove their independence, we first define the synchronous protocol in an equivalent but more convenient way.

Consider for each ordered pair $u \sim v$ a pair of exponential clocks $Z_{u,v}, Z'_{u,v}$, both with rate $1/\deg(u)$. All these clocks are independent. We say the clocks $Z_{u,v}, Z'_{u,v}$ are *located* at vertex u. Initially, the clocks $Z_{u,v}$ are turned on, and the clocks $Z'_{u,v}$ are off. For each round $1, 2, \ldots,$ we visit the vertices one by one. For each vertex u, we wait for the next clock at u to ring. If that ring comes from clock $Z_{u,v}$ or $Z'_{u,v}$, we say that u calls v in that round. Once the choice of calls at every vertex has been made, we use these to perform the push&pull operations in a round of the protocol. (Note that the time of the clocks is separate from the discrete rounds of the synchronous protocol: in each vertex, a different amount of time is elapsed on the clocks.) Moreover, whenever a vertex u gets informed of the rumor, for each adjacent v we turn off the clocks $Z_{u,v}$ and $Z_{v,u}$, and turn on $Z'_{u,v}$ and $Z'_{v,u}$. (If v was already informed, these status changes had already taken place.) Observe that, because of memorylessness of the exponential distribution, for each vertex u, this process generates a random sequence of independent uniform neighbors, so it is equivalent to the synchronous protocol.

Now let us see what are the random variables $T_{u,v}$ in this setup. For each ordered pair u, v, observe that the collection of ringing times of clocks $Z_{u,v}, Z'_{u,v}$ forms a Poisson process $P_{u,v}$ with rate $1/\deg(u)$. (It does not matter that the initial rings come from Z and subsequent rings from Z'.) Let

$$P_u := \bigcup_{v\sim u} P_{u,v},$$

and note that P_u is a Poisson process with rate 1.

For a pair u, v, suppose the q_{uv}th point in P_u is at α, and suppose the first point of $P_{u,v}$ strictly larger than α is at β. Then, $T_{u,v}$ is precisely the number of points of P_u in the interval $(\alpha, \beta]$. Define $X_{u,v} = \beta - \alpha$. By construction, $X_{u,v}$ is the first time that clock $Z'_{u,v}$ rung from the time it was turned on, hence it is exponential with rate $1/\deg(u)$. Since these clocks are independent, the random variables $X_{u,v}$ are also independent. (Only the times at which the clocks are turned on depend on other clocks in a non-trivial manner.) Thus we have proven:

▶ **Lemma 7.** *The random variables $\{X_{x,y}\}$ defined above are mutually independent.*

On the other hand, we can use these to control the $T_{x,y}$:

▶ **Lemma 8.** *For any fixed K there is a fixed C such that with probability at least $1 - n^{-K}$, for all adjacent pairs x, y we have $T_{x,y} \leq C \log n + CX_{x,y}$.*

Proof. We show that for any adjacent pair x, y, we have $T_{x,y} > C \log n + CX_{x,y}$ with probability at most n^{-K-2}, and then apply the union bound over all edges.

Observe that, conditioned on $X_{x,y} = t$, the random variable $T_{x,y} - 1$ is Poisson with rate $t \times (\deg(u) - 1)/\deg(u) \leq t$. We will use a standard tail inequality for the Poisson distribution, which follows from Theorem 5.1(iii) in [10]. Let $\mathrm{Po}(t)$ denote a Poisson random variable with mean $t > 0$. Then for any $\alpha \geq 1$ we have $\mathbb{P}[\mathrm{Po}(t) \geq \alpha t] \leq (e^{\alpha-1}\alpha^{-\alpha})^t$. This gives

$$\begin{aligned}\mathbb{P}[T_{x,y} - 1 > C \log n + CX_{x,y} | X_{x,y} = t] &\leq \mathbb{P}[\mathrm{Po}(t) > C \log n + Ct] \\ &\leq (e/C)^{C \log n} \leq n^{-K-2}\end{aligned}$$

for $C \geq \max(e^2, K + 2)$. ◀

Our main result now follows easily from our lemmas.

Proof of Theorem 3. Given K, pick C sufficiently large so that Lemmas 6 and 8 hold. Fix $t \geq 1$ and let $L = Ct^{2/3}n^{1/3}$. We have

$$\begin{aligned}&\mathbb{P}[S > Ct + CL \log n] \\ &\leq \mathbb{P}\left[\left(\max_{v \in V} \min_{\gamma \in \Gamma(s,v)} \sum_{xy \in E(\gamma)} \min\{T_{x,y}, T_{y,x}\}\right) > Ct + CL \log n\right] \\ &\leq \mathbb{P}\left[\left(\max_{v \in V} \min_{\substack{\gamma \in \Gamma(s,v) \\ |\gamma| \leq L}} \sum_{xy \in E(\gamma)} \min\{T_{x,y}, T_{y,x}\}\right) > Ct + CL \log n\right] \\ &\leq \mathbb{P}\left[\left(\max_{v \in V} \min_{\substack{\gamma \in \Gamma(s,v) \\ |\gamma| \leq L}} \sum_{xy \in E(\gamma)} C \log n + C \min\{X_{x,y}, X_{y,x}\}\right) > Ct + CL \log n\right] + n^{-K} \\ &\leq \mathbb{P}\left[\left(\max_{v \in V} \min_{\substack{\gamma \in \Gamma(s,v) \\ |\gamma| \leq L}} \sum_{xy \in E(\gamma)} C \min\{X_{x,y}, X_{y,x}\}\right) > Ct\right] + n^{-K} \\ &= \mathbb{P}[A_L > t] + n^{-K} \\ &\leq \mathbb{P}[A > t] + n^{-K} + e^{-Cn^{1/3}}.\end{aligned}$$

Here, the first inequality is copied from (4). The second inequality is because restricting the feasible region of a minimization problem can only increase its optimal value. The third inequality follows from Lemma 8. The fourth inequality is straightforward. The equality follows from the definition of A_L and noting that $\{X_{x,y}\}$ have the same joint distribution as $\{Y_{x,y}\}$, and the last inequality follows from Lemma 6. This completes the proof of Theorem 3. ◀

Acknowledgements. We would like to thank an anonymous referee for pointing out a mistake in the statement of Corollary 5 in an earlier version.

References

1 Hüseyin Acan, Andrea Collevecchio, Abbas Mehrabian, and Nick Wormald. On the push&pull protocol for rumor spreading. *SIAM Journal on Discrete Mathematics*, 31(2):647–668, 2017. Available in `https://arxiv.org/abs/1411.0948` (conference version in PODC'15). `doi:10.1137/15M1033113`.

2 A. Auffinger, M. Damron, and J. Hanson. 50 years of first passage percolation. *arXiv*, 1511.03262 [math.PR], 2016.

3 Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE Trans. Inform. Theory*, 52(6):2508–2530, 2006. `doi:10.1109/TIT.2006.874516`.

4 Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database maintenance. In *Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing*, PODC'87, pages 1–12, New York, NY, USA, 1987. ACM. `doi:10.1145/41840.41841`.

5 B. Doerr, M. Fouz, and T. Friedrich. Experimental analysis of rumor spreading in social networks. In *Design and analysis of algorithms*, volume 7659 of *Lecture Notes in Comput. Sci.*, pages 159–173. Springer, Heidelberg, 2012. `doi:10.1007/978-3-642-34862-4_12`.

6 R. Durrett. Stochastic growth models: recent results and open problems. In *Mathematical approaches to problems in resource management and epidemiology (Ithaca, NY, 1987)*, volume 81 of *Lecture Notes in Biomath.*, pages 308–312. Springer, Berlin, 1989. `doi:10.1007/978-3-642-46693-9_21`.

7 Uriel Feige, David Peleg, Prabhakar Raghavan, and Eli Upfal. Randomized broadcast in networks. *Random Structures Algorithms*, 1(4):447–460, 1990. `doi:10.1002/rsa.3240010406`.

8 G. Giakkoupis, Y. Nazari, and P. Woelfel. How asynchrony affects rumor spreading time. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, PODC'16, pages 185–194, New York, NY, USA, 2016. ACM. `doi:10.1145/2933057.2933117`.

9 J. M. Hammersley and D. J. A. Welsh. *First-Passage Percolation, Subadditive Processes, Stochastic Networks, and Generalized Renewal Theory*, pages 61–110. Springer Berlin Heidelberg, Berlin, Heidelberg, 1965.

10 S. Janson. Tail bounds for sums of geometric and exponential variables. Available in `http://www2.math.uu.se/~svante/papers/sjN14.pdf`.

11 R. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking. Randomized rumor spreading. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 565–574, 2000. `doi:10.1109/SFCS.2000.892324`.

12 D. Richardson. Random growth in a tessellation. *Proc. Cambridge Philos. Soc.*, 74:515–528, 1973.

Sharper Bounds for Regularized Data Fitting*

Haim Avron[1], Kenneth L. Clarkson[2], and David P. Woodruff[3]

1 **Tel Aviv University, Tel Aviv, Israel**
`haimav@post.tau.ac.il`

2 **IBM Research – Almaden, San Jose, CA, USA**
`klclarks@us.ibm.com`

3 **IBM Research – Almaden, San Jose, CA, USA**
`dpwoodru.ibm.com`

Abstract

We study matrix sketching methods for regularized variants of linear regression, low rank approximation, and canonical correlation analysis. Our main focus is on sketching techniques which preserve the objective function value for regularized problems, which is an area that has remained largely unexplored. We study regularization both in a fairly broad setting, and in the specific context of the popular and widely used technique of ridge regularization; for the latter, as applied to each of these problems, we show algorithmic resource bounds in which the *statistical dimension* appears in places where in previous bounds the rank would appear. The statistical dimension is always smaller than the rank, and decreases as the amount of regularization increases. In particular, for the ridge low-rank approximation problem $\min_{Y,X} \|YX - A\|_F^2 + \lambda\|Y\|_F^2 + \lambda\|X\|_F^2$, where $Y \in \mathbb{R}^{n\times k}$ and $X \in \mathbb{R}^{k\times d}$, we give an approximation algorithm needing $O(\mathtt{nnz}(A)) + \tilde{O}((n+d)\varepsilon^{-1}k\min\{k, \varepsilon^{-1}\,\mathtt{sd}_\lambda(Y^*)\}) + \mathrm{poly}(\mathtt{sd}_\lambda(Y^*)\epsilon^{-1})$ time, where $s_\lambda(Y^*) \le k$ is the statistical dimension of Y^*, Y^* is an optimal Y, ε is an error parameter, and $\mathtt{nnz}(A)$ is the number of nonzero entries of A. This is faster than prior work, even when $\lambda = 0$. We also study regularization in a much more general setting. For example, we obtain sketching-based algorithms for the low-rank approximation problem $\min_{X,Y} \|YX - A\|_F^2 + f(Y,X)$ where $f(\cdot,\cdot)$ is a regularizing function satisfying some very general conditions (chiefly, invariance under orthogonal transformations).

1998 ACM Subject Classification G.1.3 Numerical Linear Algebra

Keywords and phrases Matrices, Regression, Low-rank approximation, Regularization, Canonical Correlation Analysis

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.27

1 Introduction

The technique of matrix sketching, such as the use of random projections, has been shown in recent years to be a powerful tool for accelerating many important statistical learning techniques. Indeed, recent work has proposed highly efficient algorithms for, among other problems, linear regression, low-rank approximation [22, 30] and canonical correlation analysis [3]. In addition to being a powerful theoretical tool, sketching is also an applied one; see [31] for a discussion of state-of-the-art performance for important techniques in statistical learning.

Many statistical learning techniques can benefit substantially, in their quality of results, by using some form of regularization. Regularization can also help by reducing the computing

[*] Condensed version; most proofs omitted. Full version: `https://arxiv.org/abs/1611.03225`.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 27; pp. 27:1–27:22
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

resources needed for these techniques. While there has been some prior exploration in this area, as discussed in §1.1, commonly it has featured sampling-based techniques, often focused on regression, and often with analyses using distributional assumptions about the input (though such assumptions are not always necessary). Our study considers fast (linear-time) sketching methods, a breadth of problems, and makes no distributional assumptions. Also, where most prior work studied the distance of an approximate solution to the optimum, our guarantees are concerning approximation with respect to a relevant loss function - see below for more discussion.

It is a long-standing theme in the study of randomized algorithms that structures that aid statistical inference can also aid algorithm design, so that for example, VC dimension and sample compression have been applied in both areas, and more recently, in cluster analysis the algorithmic advantages of natural statistical assumptions have been explored. This work is another contribution to this theme. Our high-level goal in this work is to study generic conditions on sketching matrices that can be applied to a wide array of regularized problems in linear algebra, preserving their objective function values, and exploiting the power of regularization.

1.1 Results

We study regularization both in a fairly broad setting, and in the specific context of the popular and widely used technique of ridge regularization. We discuss the latter in sections 2, 3 and B; our main results for ridge regularization, Theorem 15, on linear regression, Theorem 26, on low-rank approximation, and Theorem 33, on canonical correlation analysis, show that for ridge regularization, the sketch size need only be a function of the *statistical dimension* of the input matrix, as opposed to its rank, as is common in the analysis of sketching-based methods. Thus, ridge regularization improves the performance of sketching-based methods.

Next, we consider regularizers under rather general assumptions involving invariance under left and/or right multiplication by orthogonal matrices, and show that sketching-based methods can be applied, to regularized multiple-response regression in §C and to regularized low-rank approximation, in §D. Here we obtain running times in terms of the statistical dimension. Along the way, in §D.1, we give a "base case" algorithm for reducing low-rank approximation, via singular value decomposition, to the special case of diagonal matrices.

Throughout we rely on sketching matrix constructions involving *sparse embeddings* [10, 24, 23, 6, 12], and on *Sampled Randomized Hadamard Transforms* (SRHT) [1, 26, 14, 15, 28, 7, 16, 33]. Here for matrix A, its sketch is SA, where S is a sketching matrix. The sketching constructions mentioned can be combined to yield a sketching matrix S such that the sketch of matrix A, which is simply SA, can be computed in time $O(\mathtt{nnz}(A))$, which is proportional to the number of nonzero entries of A. Moreover, the number of rows of S is small. Corollary 14 summarizes our use of these constructions as applied to ridge regression.

A key property of a sketching matrix S is that it be a *subspace embedding*, so that $\|SAx\|_2 \approx \|Ax\|_2$ for all x. Definition 20 gives the technical definition, and Definition 22 gives the definition of the related property of an *affine embedding* that we also use. Lemma 23 summarizes the use of sparse embeddings and SRHT for subspace and affine embeddings.

In the following we give our main results in more detail. However, before doing so, we need the formal definition of the statistical dimension.

▶ **Definition 1** (Statistical Dimension). For real value $\lambda \geq 0$ and rank-k matrix A with singular values $\sigma_i, i \in [k]$, the quantity $\mathtt{sd}_\lambda(A) \equiv \sum_{i\in[k]} 1/(1 + \lambda/\sigma_i^2)$ is the *statistical dimension* (or *effective dimension*, or "hat matrix trace") of the ridge regression problem with regularizing weight λ.

Note that $\mathtt{sd}_\lambda(A)$ is decreasing in λ, with maximum $\mathtt{sd}_0(A)$ equal to the rank of A. Thus a dependence of resources on $\mathtt{sd}_\lambda(A)$ instead of the rank is never worse, and will be much better for large λ.

In §A, we give an algorithm for estimating $\mathtt{sd}_\lambda(A)$ to within a constant factor, in $O(\mathtt{nnz}(A))$ time, for $\mathtt{sd}_\lambda(A) \leq (n+d)^{1/3}$. Knowing $\mathtt{sd}_\lambda(A)$ to within a constant factor allows us to set various parameters of our algorithms.

1.1.1 Ridge Regression

In §2 we apply sketching to reduce from one ridge regression problem to another one with fewer rows.

▶ **Theorem 2** (Less detailed version of Thm. 15). *Given $\varepsilon \in (0,1]$ and $A \in \mathbb{R}^{n\times d}$, there is a sketching distribution over $S \in \mathbb{R}^{m\times n}$, where $m = \tilde{O}(\varepsilon^{-1}\mathtt{sd}_\lambda(A))$, such that SA can be computed in $O(\mathtt{nnz}(A)) + d \cdot \mathrm{poly}(\mathtt{sd}_\lambda(A)/\varepsilon)$ time, and with constant probability $\tilde{x} \equiv \mathrm{argmin}_{x\in\mathbb{R}^d}\|S(Ax-b)\|^2 + \lambda\|x\|^2$ satisfies*

$$\|A\tilde{x}-b\|^2 + \lambda\|\tilde{x}\|^2 \leq (1+\varepsilon)\min_{x\in\mathbb{R}^d}\|Ax-b\|^2 + \lambda\|x\|^2.$$

Here $\mathrm{poly}(\kappa)$ denotes some polynomial function of the value κ.

In our analysis (Lemma 10), we map ridge regression to ordinary least squares (by using a matrix with $\sqrt{\lambda}I$ adjoined), and then apply prior analysis of sketching algorithms, but with the novel use of a sketching matrix that is "partly exact"; this latter step is important to obtain our overall bounds. We also show that sketching matrices can be usefully composed in our regularized setting; this is straightforward in the non-regularized case, but requires some work here.

As noted, the statistical dimension of a data matrix in the context of ridge regression is also referred to as the *effective degrees of freedom* of the regression problem in the statistics literature, and the statistical dimension features, as the name suggests, in the statistical analysis of the method. Our results show that the statistical dimension affects not only the statistical capacity of ridge regression, but also its computational complexity.

The reduction of the above theorem is mainly of interest when $n \gg \mathtt{sd}_\lambda(A)$, which holds in particular when $n \gg d$, since $d \geq \mathtt{rank}(A) \geq \mathtt{sd}_\lambda(A)$. We also give a reduction using sketching when d is large, discussed in §2.2. Here algorithmic resources depend on a power of σ_1^2/λ, where σ_1 is the leading singular value of A. This result falls within our theme of improved efficiency as λ increases, but in contrast to our other results, performance does not degrade gracefully as $\lambda \to 0$. The difficulty is that we use the product of sketches $AS^\top SA^\top$ to estimate the product $AA^\top$ in the expression $\|AA^\top y - b\|$. Since that expression can be zero, and since we seek a strong notion of relative error, the error of our overall estimate is harder to control, and impossible when $\lambda = 0$.

As for related work on ridge regression, Lu *et al.* [21] apply the SRHT to ridge regression, analyzing the statistical risk under the distributional assumption on the input data that b is a random variable, and not giving bounds in terms of $\mathtt{sd}_\lambda$. El Alaoui *et al.* [17] apply sampling techniques based on the *leverage scores* of a matrix derived from the input, with a different error measure than ours, namely, the statistical risk; here for their error analysis they consider the case when the noise in their ridge regression problem is i.i.d. Gaussian. They give results in terms of $\mathtt{sd}_\lambda(A)$, which arises naturally for them as the sum of the leverage scores. Here we show that this quantity arises also in the context of oblivious subspace embeddings, and with the goal being to obtain a worst-case relative-error guarantee in objective function value rather than for minimizing statistical risk. Chen *et al.* [9] apply sparse embeddings to ridge regression, obtaining solutions $\tilde{x}$ with $\|\tilde{x} - x^*\|_2$ small, where x^* is optimal, and do this in $O(\mathtt{nnz}(A) + d^3/\varepsilon^2)$ time. They also analyze the statistical risk of their output. Yang *et al.* [32] consider slower sketching methods than those here, and analyze their error under distributional assumptions using an incomparable notion of statistical dimension. Frostig *et al.* [18] make distributional assumptions, in particular a kurtosis property. Frostig *et al.* [19] give bounds in terms of a convex condition number that can be much larger than $\mathtt{sd}_\lambda(A)$. Another related work is that of Pilanci *et al.* [25] which we dicuss below.

1.1.2 Ridge Low-rank Approximation

In §3 we consider the following problem: for given $A \in \mathbb{R}^{n\times d}$, integer k, and weight $\lambda \geq 0$, find:

$$\min_{\substack{Y\in\mathbb{R}^{n\times k}\\ X\in\mathbb{R}^{k\times d}}} \|YX - A\|_F^2 + \lambda\|Y\|_F^2 + \lambda\|X\|_F^2, \tag{1}$$

where, as is well known (and discussed in detail later), this regularization term is equivalent to $2\lambda\|YX\|_*$, where $\|\cdot\|_*$ is the trace (nuclear) norm, the Schatten 1-norm. We show the following.

▶ **Theorem 3** (Less detailed Thm. 26). *Given input $A \in \mathbb{R}^{n\times d}$, there is a sketching-based algorithm returning $\tilde{Y} \in \mathbb{R}^{n\times k}, \tilde{X} \in \mathbb{R}^{k\times d}$ such that with constant probability, $\tilde{Y}$ and $\tilde{X}$ form a $(1+\varepsilon)$-approximate minimizer to (1), that is,*

$$\|\tilde{Y}\tilde{X} - A\|_F^2 + \lambda\|\tilde{Y}\|_F^2 + \lambda\|\tilde{X}\|_F^2 \tag{2}$$

$$\leq (1+\varepsilon) \min_{\substack{Y\in\mathbb{R}^{n\times k}\\ X\in\mathbb{R}^{k\times d}}} \|YX - A\|_F^2 + \lambda\|Y\|_F^2 + \lambda\|X\|_F^2. \tag{3}$$

The matrices $\tilde{Y}$ and $\tilde{X}$ can be found in $O(\mathtt{nnz}(A)) + \tilde{O}((n+d)\varepsilon^{-1}k\min\{k, \varepsilon^{-1}\,\mathtt{sd}_\lambda(Y^*)\}) + \mathrm{poly}(\varepsilon^{-1}\,\mathtt{sd}_\lambda(Y^*))$ time, where Y^* is an optimum Y in (1) such that $\mathtt{sd}_\lambda(X^*) = \mathtt{sd}_\lambda(Y^*) \leq \mathtt{rank}(Y^*) \leq k$.

This algorithm follows other algorithms for $\lambda = 0$ with running times of the form $O(\mathtt{nnz}(A)) + (n+d)\mathrm{poly}(k/\varepsilon)$ (e.g. [10]), and has the best known dependence on k and ε for algorithms of this type, even when $\lambda = 0$.

Our approach is to first extend our ridge regression results to the multiple-response case $\min_Z \|AZ - B\|_F^2 + \lambda\|Z\|_F^2$, and then reduce the multiple-response problem to a smaller one by showing that up to a cost in solution quality, we can assume that each row of Z lies in the rowspace of SA, for S a suitable sketching matrix. We apply this observation twice to the low-rank approximation problem, so that Y can be assumed to be of the form $AR\tilde{Y}$, and X of the form $\tilde{X}SA$, for sketching matrix S and (right) sketching matrix R. Another round of sketching then reduces to a low-rank approximation problem of size independent of n and d, and finally an SVD-based method is applied to that small problem.

Regarding related work: the regularization "encourages" the rank of YX to be small, even when there is no rank constraint (k is large), and this unconstrained problem has been extensively studied; even so, the rank constraint can reduce the computational cost and improve the output quality, as discussed by [8], who also give further background, and who give experimental results on an iterative algorithm. Pilanci *et al.* [25] consider only algorithms where the sketching time is at least $\Omega(nd)$, which can be much slower than our $\mathtt{nnz}(A)$ for sparse matrices, and it is not clear if their techniques can be extended. In the case of low-rank approximation with a nuclear norm constraint (the closest to our work), as the authors note, their paper gives no improvement in running time. While their framework might imply analyses for ridge regression, they did not consider it specifically, and such an analysis may not follow directly.

1.1.3 Regularized Canonical Correlation Analysis

Canonical correlation analysis (CCA) is an important statistical technique whose input is a pair of matrices, and whose solution depends on the Gram matrices $A^\top A$ and $B^\top B$. If these Gram matrices are ill-conditioned it is useful to regularize them by instead using $A^\top A + \lambda_1 I_d$ and $B^\top B + \lambda_2 I_{d'}$, for weights $\lambda_1, \lambda_2 \geq 0$. Thus, in this paper we consider a regularized version of CCA, defined as follows (our definition is in the same spirit as the one used by [3]).

▶ **Definition 4.** Let $A \in \mathbb{R}^{n\times d}$ and $B \in \mathbb{R}^{n\times d'}$, and let

$$q = \min(\texttt{rank}(A^\top A + \lambda_1 I_d), \texttt{rank}(B^\top B + \lambda_2 I_{d'})).$$

Let $\lambda_1 \geq 0$ and $\lambda_2 \geq 0$. The (λ_1, λ_2) *canonical correlations* $\sigma_1^{(\lambda_1,\lambda_2)} \geq \cdots \geq \sigma_q^{(\lambda_1,\lambda_2)}$ and (λ_1, λ_2) *canonical weights* $u_1, \ldots, u_q \in \mathbb{R}^d$ and $v_1, \ldots, v_q \in \mathbb{R}^{d'}$ are ones that maximize

$$\texttt{tr}(U^\top A^\top B V)$$

subject to

$$\begin{aligned} U^\top (A^\top A + \lambda_1 I_d) U &= I_q \\ V^\top (B^\top B + \lambda_2 I_{d'}) V &= I_q \\ U^\top A^\top B V &= \texttt{diag}(\sigma_1^{(\lambda_1,\lambda_2)}, \ldots, \sigma_q^{(\lambda_1,\lambda_2)}) \end{aligned}$$

where $U = [u_1, \ldots, u_q] \in \mathbb{R}^{n\times q}$ and $V = [v_1, \ldots, v_q] \in \mathbb{R}^{d'\times q}$.

One classical way to solve non-regularized CCA ($\lambda_1 = \lambda_2 = 0$) is the Björck-Golub algorithm [5]. In §B we show that regularized CCA can be solved using a variant of the Björck-Golub algorithm.

Avron et al. [3] showed how to use sketching to compute an approximate CCA. In §B we show how to use sketching to compute an approximate regularized CCA.

▶ **Theorem 5** (Loose version of Thm. 33). *There is a distribution over matrices $S \in \mathbb{R}^{m\times n}$ with $m = O(\max(\texttt{sd}_{\lambda_1}(A), \texttt{sd}_{\lambda_2}(B))^2/\epsilon^2)$ such that with constant probability, the regularized CCA of (SA, SB) is an ϵ-approximate CCA of (A, B). The matrices SA and SB can be computed in $O(\texttt{nnz}(A) + \texttt{nnz}(B))$ time.*

Our generalization of the classical Björck-Golub algorithm shows that regularized canonical correlation analysis can be computed via the product of two matrices whose columns are non-orthogonal regularized bases of A and B. We then show that these two matrices are easier to sketch than the orthogonal bases that arise in non-regularized CCA. This in turn can be tied to approximation bounds of sketched regularized CCA versus exact CCA.

1.1.4 General Regularization

A key property of the Frobenius norm $\|\cdot\|_F$ is that it is invariant under rotations; for example, it satisfies the *right orthogonal invariance* condition $\|AQ\|_F = \|A\|_F$, for any orthogonal matrix Q (assuming, of course, that A and Q having dimensions so that AQ is defined). In

§C and §D, we study conditions under which such an invariance property, and little else, is enough to allow fast sketching-based approximation algorithms.

For regularized multiple-response regression, we have the following.

▶ **Theorem 6** (Implied by Thm. 39). *Let $f(\cdot)$ be a real-valued function on matrices that is right orthogonally invariant, subadditive, and invariant under padding the input matrix by rows or columns of zeros. Let $A \in \mathbb{R}^{n \times d}, B \in \mathbb{R}^{n \times d'}$. Suppose that for $r \equiv \texttt{rank}\, A$, there is an algorithm that for general n, d, d', r and $\varepsilon > 0$, in time $\tau(d, n, d', r, \varepsilon)$ finds $\tilde{X}$ with*

$$\|A\tilde{X} - B\|_F^2 + f(\tilde{X}) \leq (1 + \varepsilon) \min_{X \in \mathbb{R}^{d \times d'}} \|AX - B\|_F^2 + f(X).$$

Then there is another algorithm that with constant probability finds such an $\tilde{X}$, taking time

$$O(\texttt{nnz}(A) + \texttt{nnz}(B) + (n + d + d')\text{poly}(r/\varepsilon)) + \tau(d, \text{poly}(r/\varepsilon), \text{poly}(r/\varepsilon), r, \varepsilon).$$

That is, sketching can be used to reduce to a problem in which the only remaining large matrix dimension is d, the number of columns of A.

This reduction is a building block for our results for regularized low-rank approximation. Here the regularizer is a real-valued function $f(Y, X)$ on matrices $Y \in \mathbb{R}^{n \times k}, X \in \mathbb{R}^{k \times d}$. We show that under broad conditions on $f(\cdot, \cdot)$, sketching can be applied to

$$\min_{\substack{Y \in \mathbb{R}^{n \times k} \\ X \in \mathbb{R}^{k \times d}}} \|YX - A\|_F^2 + f(Y, X). \tag{4}$$

Our conditions imply fast algorithms when, for example, $f(Y, X) = \|YX\|_{(p)}$, where $\|\cdot\|_{(p)}$ is a Schatten p-norm, or when $f(Y, X) = \min\{\lambda_1 \|YX\|_{(1)}, \lambda_2 \|YX\|_{(2)}\}$, for weights λ_1, λ_2, and more. Of course, there are norms, such as the entriwise ℓ_1 norm, that do not satisfy these orthogonal invariance conditions.

▶ **Theorem 7** (Implied by Thm. 44). *Let $f(Y, X)$ be a real-valued function on matrices that in each argument is subadditive and invariant under padding by rows or columns of zeros, and also right orthogonally invariant in its right argument and left orthogonally invariant in its left argument.*

Suppose there is a procedure that solves (4) when A, Y, and X are $k \times k$ matrices, and A is diagonal, and YX is constrained to be diagonal, taking time $\tau(k)$ for a function $\tau(\cdot)$.

Then for general A, there is an algorithm that finds a $(1 + \varepsilon)$-approximate solution $(\tilde{Y}, \tilde{X})$ in time $O(\texttt{nnz}(A)) + \tilde{O}(n + d)\text{poly}(k/\varepsilon) + \tau(k)$.

The proof involves a reduction to small matrices, followed by a reduction, discussed in §D.1, that uses the SVD to reduce to the diagonal case. This result, Corollary 43, generalizes results of [29], who gave such a reduction for $f(Y, X) = \|X\|_F^2 + \|Y\|_F^2$; also, we give a very different proof.

As for related work, [29] survey and extend work in this setting, and propose iterative algorithms for this problem. The regularizers $f(Y, X)$ they consider, and evaluate experimentally, are more general than we can analyze.

The conditions on $f(Y, X)$ are quite general; it may be that for some instances, the resulting problem is NP-hard. Here our reduction would be especially interesting, because the size of the reduced NP-hard problem depends only on k.

1.2 Basic Definitions and Notation

We denote scalars using Greek letters. Vectors are denoted by $x, y, \ldots$ and matrices by $A, B, \ldots$. We use the convention that vectors are column-vectors. We use $\texttt{nnz}(\cdot)$ to denote the number of nonzeros in a vector or matrix. We denote by $[n]$ the set $\{1, \ldots, n\}$. The notation $\alpha = (1 \pm \gamma)\beta$ means that $(1-\gamma)\beta \le \alpha \le (1+\gamma)\beta$. Throughout the paper, A denotes an $n \times d$ matrix, and $\sigma_1 \ge \sigma_2 \ge \cdots \ge \sigma_{\min(n,d)}$ its singular values.

▶ **Definition 8** (Schatten p-norm)**.** The *Schatten p-norm* of A is $\|A\|_{(p)} \equiv [\sum_i \sigma_i^p]^{1/p}$. Note that the trace (nuclear) norm $\|A\|_* = \|A\|_{(1)}$, the Frobenius norm $\|A\|_F = \|A\|_{(2)}$, and the spectral norm $\|A\|_2 = \|A\|_{(\infty)}$.

The notation $\|\cdot\|$ without a subscript denotes the ℓ_2 norm for vectors, and the spectral norm for matrices. We use a subscript for other norms. We use $\texttt{range}(A)$ to denote the subspace spanned by the columns of A, i.e. $\texttt{range}(A) \equiv \{Ax \mid x \in \mathbb{R}^d\}$. I_d denotes the $d \times d$ identity matrix, 0_d denotes the column vector comprising d entries of zero, and $0_{a\times b} \in \mathbb{R}^{a\times b}$ denotes a zero matrix.

The rank $\texttt{rank}(A)$ of a matrix A is the dimension of the subspace $\texttt{range}(A)$ spanned by its columns (equivalently, the number of its non-zero singular values). Bounds on sketch sizes are often written in terms of the rank of the matrices involved.

▶ **Definition 9** (Stable Rank)**.** The *stable rank* $\texttt{sr}(A) \equiv \|A\|_F^2/\|A\|_2^2$. The stable rank satisfies $\texttt{sr}(A) \le \texttt{rank}(A)$.

Paper Outline: Due to space constraints, most proofs are omitted, and all results except our results for ridge regression and ridge low-rank approximation are deferred to the appendix. The missing proofs and results can also be found in the full version of our paper on arXiv under the same title: `https://arxiv.org/abs/1611.03225`.

2 Ridge Regression

Let $A \in \mathbb{R}^{n\times d}$, $b \in \mathbb{R}^n$, and $\lambda > 0$. In this section we consider the *ridge regression* problem:

$$\min_{x\in\mathbb{R}^d} \|Ax - b\|^2 + \lambda\|x\|^2, \tag{5}$$

Let $x^* \equiv \mathrm{argmin}_{x\in\mathbb{R}^d}\|Ax-b\|^2 + \lambda\|x\|^2$ and $\Delta_* \equiv \|Ax^* - b\|^2 + \lambda\|x^*\|^2$. In general $x^* = (A^\top A + \lambda I_d)^{-1}A^\top b = A^\top(AA^\top + \lambda I_n)^{-1}b$, so $x^\star$ can be found in $O(\texttt{nnz}(A)\min(n,d))$ time using an iterative method (e.g., LSQR). Our goal in this section is to design faster algorithms that find an approximate $\tilde{x}$ in the following sense:

$$\|A\tilde{x} - b\|^2 + \lambda\|\tilde{x}\|^2 \le (1+\varepsilon)\Delta_* \,. \tag{6}$$

In our analysis, we distinguish between two cases: $n \gg d$ and $d \gg n$.

▶ Remark. In this paper we consider only approximations of the form (6). Although we do not explore it in this paper, our techniques can also be used to derive preconditioned methods. Analysis of preconditioned kernel ridge regression, which is related to the $d \gg n$ case, is explored in [4].

2.1 Large n

In this subsection we design an algorithm that is aimed at the case when $n \gg d$. However, the results themselves are correct even when $n < d$. The general strategy is to design a distribution on matrices of size m-by-n (m is a parameter), sample an S from that distribution, and solve $\tilde{x} \equiv \operatorname{argmin}_{x \in \mathbb{R}^d} \|S(Ax - b)\|^2 + \lambda \|x\|^2$.

The following lemma defines conditions on the distribution that guarantee that (6) holds with constant probability (which can be boosted to high probability by repetition and taking the solution with minimum objective value).

▶ **Lemma 10.** *Let $x^* \in \mathbb{R}^d$, A and b as above. Let $U_1 \in \mathbb{R}^{n \times d}$ comprise the first n rows of an orthogonal basis for $\left[\begin{smallmatrix} A \\ \sqrt{\lambda} I_d \end{smallmatrix}\right]$. Let sketching matrix $S \in \mathbb{R}^{m \times n}$ have a distribution such that with constant probability*

$$\|U_1^\top S^\top S U_1 - U_1^\top U_1\|_2 \leq 1/4, \tag{7}$$

and

$$\|U_1^\top S^\top S(b - Ax^*) - U_1^\top (b - Ax^*)\| \leq \sqrt{\varepsilon \Delta_* / 2}. \tag{8}$$

Then with constant probability, $\tilde{x} \equiv \operatorname{argmin}_{x \in \mathbb{R}^d} \|S(Ax-b)\|^2 + \lambda \|x\|^2$ has $\|A\tilde{x} - b\|^2 + \lambda \|\tilde{x}\|^2 \leq (1 + \varepsilon)\Delta_*$.

Proof. Omitted in this version. ◀

▶ **Lemma 11.** *For U_1 as in Lemma 10, $\|U_1\|_F^2 = \mathtt{sd}_\lambda(A) = \sum_i 1/(1 + \lambda/\sigma_i^2)$, where A has singular values σ_i. Also $\|U_1\|_2 = 1/\sqrt{1 + \lambda/\sigma_1^2}$.*

This follows from (3.47) of [20]; for completeness, a proof is given here.

Proof. Suppose $A = U\Sigma V^\top$, the full SVD, so that $U \in \mathbb{R}^{n \times n}$, $\Sigma \in \mathbb{R}^{n \times d}$, and $V \in \mathbb{R}^{d \times d}$. Let $D \equiv (\Sigma^\top \Sigma + \lambda I_d)^{-1/2}$. Then $\hat{A} = \left[\begin{smallmatrix} U\Sigma D \\ V\sqrt{\lambda} D \end{smallmatrix}\right]$ has $\hat{A}^\top \hat{A} = I_d$, and for given x, there is $y = D^{-1} V^\top x$ with $\hat{A} y = \left[\begin{smallmatrix} A \\ \sqrt{\lambda} I_d \end{smallmatrix}\right] x$. We have $\|U_1\|_F^2 = \|U\Sigma D\|_F^2 = \|\Sigma D\|_F^2 = \sum_i 1/(1 + \lambda/\sigma_i^2)$ as claimed. Also $\|U_1\|_2 = \|U\Sigma D\|_2 = \|\Sigma D\|_2 = 1/\sqrt{1 + \lambda/\sigma_1^2}$, and the lemma follows. ◀

▶ **Definition 12** (large λ)**.** Say that λ is *large* for A with largest singular value σ_1, and error parameter ε, if $\lambda/\sigma_1^2 \geq 1/\varepsilon$.

The following lemma implies that if λ is large, then $x = 0$ is a good approximate solution, and so long as we include a check that a proposed solution is no worse than $x = 0$, we can assume that λ is not large.

▶ **Lemma 13.** *For $\varepsilon \in (0, 1]$, large λ, and all x, $\|Ax - b\|^2 + \lambda \|x\|^2 \geq \|b\|^2/(1 + \varepsilon)$. If λ is not large then $\|U_1\|_2^2 \geq \varepsilon/2$.*

Proof. If $\sigma_1 \|x\| \geq \|b\|$, then $\lambda \|x\|^2 \geq \sigma_1^2 \|x\|^2 \geq \|b\|^2$. Suppose $\sigma_1 \|x\| \leq \|b\|$. Then:

$$\begin{aligned}
\|Ax - b\|^2 + \lambda \|x\|^2 &= \|Ax\|^2 + \|b\|^2 - 2b^\top Ax + \lambda \|x\|^2 & \\
&\geq (\|b\| - \|Ax\|)^2 + \lambda \|x\|^2 & \text{Cauchy-Schwartz} \\
&\geq (\|b\| - \sigma_1 \|x\|)^2 + \lambda \|x\|^2 & \text{assumption} \\
&\geq \|b\|^2/(1 + \sigma_1^2/\lambda) & \text{calculus} \\
&\geq \|b\|^2/(1 + \varepsilon), & \text{large } \lambda
\end{aligned}$$

as claimed. The last statement follows from Lemma 11. ◀

Below we discuss possibilities for choosing the sketching matrix S. We want to emphasize that the first condition in Lemma 10 is *not* a subspace embedding guarantee, despite having superficial similarity. Indeed, notice that the columns of U_1 are not orthonormal, since we only take the first n rows of an orthogonal basis of $\left[\begin{smallmatrix} A \\ \sqrt{\lambda} I_d \end{smallmatrix}\right]$. Rather, the first condition is an instance of approximate matrix product with a spectral norm guarantee with constant error, for which optimal bounds in terms of the stable rank $\mathtt{sr}(U_1)$ were recently obtained [13]. As we discuss in the proof of part (i) of Corollary 14 below, $\mathtt{sr}(U_1)$ is upper bounded by $\mathtt{sd}_\lambda(A)/\epsilon$.

We only mention a few possibilities of sketching matrix S below, though others are possible with different tradeoffs and compositions.

▶ **Corollary 14.** *Suppose λ is not large (Def. 12). There is a constant $K > 0$ such that for*

(i) *$m \geq K(\varepsilon^{-1}\,\mathtt{sd}_\lambda(A) + \mathtt{sd}_\lambda(A)^2)$ and $S \in \mathbb{R}^{m\times n}$ a sparse embedding matrix (see [10, 23, 24]) with SA computable in $O(\mathtt{nnz}(A))$ time, or one can choose $m \geq K(\varepsilon^{-1}\,\mathtt{sd}_\lambda(A) + \min((\mathtt{sd}_\lambda(A)/\epsilon)^{1+\gamma}, \mathtt{sd}_\lambda(A)^2))$ an OSNAP (see [24, 6, 12]) with SA computable in $O(\mathtt{nnz}(A))$ time, where $\gamma > 0$ is an arbitrarily small constant, or*

(ii) *$m \geq K\varepsilon^{-1}(\mathtt{sd}_\lambda(A) + \log(1/\varepsilon))\log(\mathtt{sd}_\lambda(A)/\varepsilon)$ and $S \in \mathbb{R}^{m\times n}$ a Subsampled Randomized Hadamard Transform (SRHT) embedding matrix (see, e.g., [7]), with SA computable in $O(nd\log n)$ time, or*

(iii) *$m \geq K\varepsilon^{-1}\,\mathtt{sd}_\lambda(A)$ and $S \in \mathbb{R}^{m\times n}$ a matrix of i.i.d. subgaussian values with SA computable in $O(ndm)$ time,*

the conditions (7) *and* (8) *of Lemma 10 apply, and with constant probability the corresponding $\tilde{x} = \operatorname{argmin}_{x\in\mathbb{R}^d}\|S(Ax-b)\| + \lambda\|x\|^2$ is an ε-approximate solution to $\min_{x\in\mathbb{R}^d}\|b-Ax\|^2 + \lambda\|x\|^2$.*

Proof. Recall that $\mathtt{sd}_\lambda(A) = \|U_1\|_F^2$. For (i): sparse embedding distributions satisfy the bound for matrix multiplication

$$\|W^\top S^\top SH - W^\top H\|_F \leq C\|W\|_F\|H\|_F/\sqrt{m},$$

for a constant C [10, 23, 24]; this is also true of OSNAP matrices. We set $W = H = U_1$ and use $\|X\|_2 \leq \|X\|_F$ for all X and $m \geq K\|U_1\|_F^4$ to obtain (7), and set $W = U_1$, $H = b - Ax^*$ and use $m \geq K\|U_1\|_F^2/\varepsilon$ to obtain (8). (Here the bound is slightly stronger than (8), holding for $\lambda = 0$.) With (7) and (8), the claim for $\tilde{x}$ from a sparse embedding follows using Lemma 10.

For OSNAP, Theorem 1 in [13] together with [24] imply that for $m = O(\mathtt{sr}(U_1)^{1+\gamma})$, condition (7) holds. Here $\mathtt{sr}(U_1) = \frac{\|U_1\|_F^2}{\|U_1\|_2^2}$, and by Lemma 11 and Lemma 13, $\mathtt{sr}(U_1) \leq \mathtt{sd}_\lambda(A)/\epsilon$. We note that (8) continues to hold as in the previous paragraph. Thus, m is at most the min of $O((\mathtt{sd}_\lambda(A)/\epsilon)^{1+\gamma})$ and $O(\mathtt{sd}_\lambda(A)/\epsilon + \mathtt{sd}_\lambda(A)^2)$.

For (ii): Theorems 1 and 9 of [13] imply that for $\gamma \leq 1$, with constant probability

$$\|W^\top S^\top SH - W^\top H\|_2 \leq \gamma\|W\|_2\|H\|_2 \tag{9}$$

for SRHT S, when

$$m \geq C(\mathtt{sr}(W) + \mathtt{sr}(H) + \log(1/\gamma))\log(\mathtt{sr}(W) + \mathtt{sr}(H))/\gamma^2$$

for a constant C. We let $W = H = U_1$ and $\gamma = \min\{1, 1/4\|U_1\|^2\}$. We have

$$\|U_1^\top S^\top SU_1 - U_1^\top U_1\|_2 \leq \min\{1, 1/4\|U_1\|^2\}\|U_1\|_2^2 = \min\{\|U_1\|_2^2, 1/4\} \leq 1/4,$$

and

$$\mathtt{sr}(U_1)/\gamma^2 = \frac{\|U_1\|_F^2}{\|U_1\|_2^2}\max\{1, 4\|U_1\|_2^2\} = \|U_1\|_F^2 \max\{1/\|U_1\|_2^2, 4\} \le 2\|U_1\|_F^2/\varepsilon$$

using Lemma 13 and the assumption that λ is large. (And assuming $\varepsilon \le 1/2$.) Noting that $\log(1/\gamma) = O(\log(1/\varepsilon))$ and $\log(\mathtt{sr}(U_1)) = O(\log\|U_1\|_F/\varepsilon)$ using Lemma 13, we have that m as claimed suffices for (7).

For (8), we use (9) with $W = U_1$, $H = Ax^* - b$, and $\gamma = \sqrt{\varepsilon/2}/\|U_1\|_2$; note that using Lemma 13 and by the assumption that λ is large, $\gamma \le 1$ and so (9) can be applied. We have

$$\|U_1^\top S^\top S(Ax^* - b)\| \le (\sqrt{\varepsilon/2}/\|U_1\|_2)\|U_1\|_2\|Ax^* - b\| \le \sqrt{\varepsilon\Delta_*/2},$$

and

$$\mathtt{sr}(U_1)\log(\mathtt{sr}(U_1))/\gamma^2 \le \frac{\|U_1\|_F^2}{\|U_1\|_2^2}[2\log(\|U_1\|_F/\varepsilon)][2\|U_1\|_2^2/\varepsilon] = 4\|U_1\|_F^2\log(\|U_1\|_F/\varepsilon)/\varepsilon.$$

Noting that since $Ax^* - b$ is a vector, its stable rank is one, we have that m as claimed suffices for (8). With (7) and (8), the claim for $\tilde{x}$ from an SRHT follows using Lemma 10.

The claim for (iii) follows as (ii), with a slightly simpler expression for m. ◀

Here we mention the specific case of composing a sparse embedding matrix with an SRHT.

▶ **Theorem 15.** *Given $A \in \mathbb{R}^{n\times d}$, there are dimensions within constant factors of those given in Cor. 14 such that for S_1 a sparse embedding and S_2 an SRHT with those dimensions,*

$$\tilde{x} \equiv \operatorname*{argmin}_{x\in\mathbb{R}^d}\|S_2S_1(Ax - b)\|^2 + \lambda\|x\|^2,$$

satisfies $\|A\tilde{x} - b\|^2 + \lambda\|\tilde{x}\|^2 \le (1+\varepsilon)\min_{x\in\mathbb{R}^d}\|Ax - b\|^2 + \lambda\|x\|^2$ with constant probability.

Therefore in $O(\mathtt{nnz}(A)) + \tilde{O}(d\,\mathtt{sd}_\lambda(A)/\varepsilon + \mathtt{sd}_\lambda(A)^2)$ time, a ridge regression problem with n rows can be reduced to one with $O(\varepsilon^{-1}(\mathtt{sd}_\lambda(A) + \log(1/\varepsilon))\log(\mathtt{sd}_\lambda(A)/\varepsilon))$ rows, whose solution is a $(1+\varepsilon)$-approximate solution.

Proof. This follows from Corollary 14 and the general comments of Appendix A.3 of [13]; the results there imply that $\|S_iU_1\|_F = \Theta(\|U_1\|_F)$ and $\|S_iU_1\|_2 = \Theta(\|U_1\|_2)$ for $i \in [3]$ with constant probability, which implies that $\mathtt{sr}(S_1U_1)$ and $\mathtt{sr}(S_2S_1U_1)$ are $O(\mathtt{sr}(U_1))$. Moreover, the approximate multiplication bounds of (7) and (8) have versions when using $S_2S_1U_1$ and $S_2S_1(Ax^* - b)$ to estimate products involving S_1U_1 and $S_1(Ax^* - b)$, so that for example, using the triangle inequality,

$$\begin{aligned}\|U_1^\top S_1^\top S_2^\top S_2S_1U_1 - U_1^\top U_1\|_2 &\le \|U_1^\top S_1^\top S_2^\top S_2S_1U_1 - U_1^\top S_1^\top S_1U_1\|_2\\ &\quad + \|U_1^\top S_1^\top S_1U_1 - U_1^\top U_1\|_2\\ &\le 1/8 + 1/8 = 1/4.\end{aligned}$$

We have that $S = S_2S_1$ satisfies (7) and (8), as desired. ◀

Similar arguments imply that a reduction also using a sketching matrix S_3 with subgaussian entries could be used, to reduce to a ridge regression problem with $O(\varepsilon^{-1}\,\mathtt{sd}_\lambda(A))$ rows.

2.2 Large d

If the number of columns is larger than the number of rows, it is more attractive to sketch the rows, i.e., to use $AS^\top$. In general, we can express (5) as $\min_{x\in\mathbb{R}^d}\|Ax\|^2-2b^\top Ax+\|b\|^2+\lambda\|x\|^2$. We can assume x has the form $x = A^\top y$, yielding the equivalent problem

$$\min_{y\in\mathbb{R}^n}\|AA^\top y\|^2 - 2b^\top AA^\top y + \|b\|^2 + \lambda\|A^\top y\|^2. \tag{10}$$

Sketching $A^\top$ with S in the first two terms yields

$$\tilde{y} \equiv \operatorname*{argmin}_{y\in\mathbb{R}^n} \lambda\|SA^\top y\|^2 + \|AS^\top SA^\top y\|^2 - 2b^\top AA^\top y + \|b\|^2 \tag{11}$$

Now let $c^\top \equiv b^\top AA^\top$. Note that we can compute c in $O(\mathtt{nnz}(A))$ time. The solution to (11) is, for $B \equiv SA^\top$ with $B^\top B$ invertible, $\tilde{y} = (\lambda B^\top B + B^\top BB^\top B)^+ c/2$.

In the main result of this subsection, we show that provided $\lambda > 0$ then a sufficiently tight subspace embedding to $\mathtt{range}(A^\top)$ suffices.

▶ **Theorem 16.** *Suppose A has rank k, and its SVD is $A = U\Sigma V^\top$, with $U \in \mathbb{R}^{n\times k}$, $\Sigma \in \mathbb{R}^{k\times k}$ and $V \in \mathbb{R}^{d\times k}$. If $S \in \mathbb{R}^{m\times d}$ has*

1. *(Subspace Embedding) $E \equiv V^\top S^\top SV - I_k$ with $\|E\|_2 \le \varepsilon/2$*
2. *(Spectral Norm Approximate Matrix Product) for any fixed matrices C, D, each with d rows,*

$$\|C^T S^T SD - C^T D\|_2 \le \varepsilon'\|C\|_2\|D\|_2,$$

where $\varepsilon' \equiv (\varepsilon/2)/(1 + 3\sigma_1^2/\lambda)$.

Then (11) *has $\tilde{x} \equiv A^\top\tilde{y}$ approximately solving* (5)*, that is, $\|A\tilde{x} - b\|^2 + \lambda\|\tilde{x}\|^2 \le (1+\varepsilon)\Delta_*$.*

Proof. To compare the sketched with the unsketched formulations, let A have full SVD $A = U\Sigma V^\top$, and let $w = \Sigma U^\top y$. Using $\|Uz\| = \|z\|$ and $\|Vw\| = \|w\|$ yields the unsketched problem

$$\min_{w\in\mathbb{R}^k}\|\Sigma w\|^2 - 2b^\top AVw + \|b\|^2 + \lambda\|w\|^2, \tag{12}$$

equivalent to (10). The corresponding sketched version is

$$\min_{w\in\mathbb{R}^k}\|\Sigma V^\top S^\top SVw\|^2 - 2b^\top AVw + \|b\|^2 + \lambda\|SVw\|^2.$$

Now suppose S has E satisfying the first property in the theorem statement. This implies S is an $\varepsilon/2$-embedding for V:

$$|\|SVw\|^2 - \|w\|^2| = |w^\top(V^\top S^\top SV - I_k)w| \le (\varepsilon/2)\|w\|^2,$$

and, using the second property in the theorem statement with $C^T = \Sigma V^T$ and $D = V$ (which do not depend on w),

$$\|\Sigma V^\top S^\top SV - \Sigma\|_2 = f,$$

where f satisfies $|f| \le \varepsilon'\sigma_1$. It follows by the triangle inequality for any w that

$$\|\Sigma V^\top S^\top SVw\| \in [\|\Sigma w\| - f\|w\|, \|\Sigma w\| + f\|w\|].$$

Hence,

$$\begin{aligned}
|\|\Sigma V^\top S^\top S V w\|^2 - \|\Sigma w\|^2| &\in |(\|\Sigma w\| \pm f\|w\|)^2 - \|\Sigma w\|^2| \\
&\leq 2f\|\Sigma w\|\|w\| + f^2\|w\|^2 \\
\\
&\leq 3\varepsilon' \sigma_1^2 \|w\|^2
\end{aligned}$$

The value of (12) is at least $\lambda\|w\|^2$, so the relative error of the sketch is at most

$$\frac{\lambda(\varepsilon/2)\|w\|^2 + 3\varepsilon'\sigma_1^2\|w\|^2}{\lambda\|w\|^2} \leq \varepsilon.$$

The statement of the theorem follows. ◀

We now discuss which matrices S can be used in Theorem 16. Note that the first property is just the oblivious subspace embedding property, and we can use CountSketch, Subsampled Randomized Hadamard Transform, or Gaussian matrices to achieve this. One can also use OSNAP matrices [24]; note that here, unlike for Corollary 14, the running time will be $O(\mathtt{nnz}(A)/\epsilon)$ (see, e.g., [30] for a survey). For the second property, we use the recent work of [13], where tight bounds for a number of oblivious subspace embeddings S were shown.

In particular, applying the result in Appendix A.3 of [13], it is shown that the *composition* of matrices each satisfying the second property, results in a matrix also satisfying the second property. It follows that we can let S be of the form $\Pi \cdot \Pi'$, where Π' is an $r \times d$ CountSketch matrix, where $r = O(n^2/(\epsilon')^2)$, and Π is an $\tilde{O}(n/(\epsilon')^2) \times r$ Subsampled Randomized Hadamard Transform. By standard results on oblivious subspace embeddings, the first property of Theorem 16 holds provided $r = \Theta(n^2/\epsilon^2)$ and Π has $\tilde{O}(n/\epsilon^2)$ rows. Note that $\epsilon' \leq \epsilon$, so in total we have $O(n/(\epsilon')^2)$ rows.

Thus, we can compute $B = \Pi \cdot \Pi' A^T$ in $O(\mathtt{nnz}(A)) + \tilde{O}(n^3/(\epsilon')^2)$ time, and B has $\tilde{O}(n/(\epsilon')^2)$ rows and n columns. We can thus compute $\tilde{y}$ as above in $\tilde{O}(n^3/(\epsilon')^2)$ additional time. Therefore in $O(\mathtt{nnz}(A)) + \tilde{O}(n^3/(\epsilon')^2)$ time, we can solve the problem of (5).

We note that, using our results in Section 2.1, in particular Theorem 15, we can first replace n in the above time complexities with a function of $\mathtt{sd}_\lambda(A)$ and ε, which can further reduce the overall time complexity.

2.3 Multiple-response Ridge Regression

In multiple-response ridge regression one is interested in finding $X^* \equiv \operatorname{argmin}_{X \in \mathbb{R}^{d \times d'}} \|AX - B\|_F^2 + \lambda\|X\|_F^2$, where $B \in \mathbb{R}^{n \times d'}$. It is straightforward to extend the results and algorithms for large n to multiple regression. Since we use these results when we consider regularized low-rank approximation, we state them next. The proofs are omitted as they are entirely analogous to the proofs in subsection 2.1.

▶ **Lemma 17.** *Let A, U_1, U_2 as in Lemma 10, $B \in \mathbb{R}^{n \times d'}$,*

$$X^* \equiv \operatorname*{argmin}_{X \in \mathbb{R}^{d \times d'}} \|AX - B\|_F^2 + \lambda\|X\|_F^2,$$

and $\Delta_* \equiv \|AX^* - B\|_F^2 + \lambda\|X^*\|_F^2$. Let sketching matrix $S \in \mathbb{R}^{m \times n}$ have a distribution such that with constant probability,

$$\|U_1^\top S^\top S U_1 - U_1^\top U_1\|_2 \leq 1/4, \tag{13}$$

and

$$\|U_1^\top S^\top S(B - AX^*) - U_1^\top (B - AX^*)\|_F \leq \sqrt{\varepsilon \Delta_*}. \tag{14}$$

Then with constant probability,

$$\tilde{X} \equiv \operatorname*{argmin}_{X \in \mathbb{R}^{d \times d'}} \|S(AX - B)\|_F^2 + \lambda \|X\|_F^2 \tag{15}$$

has $\|A\tilde{X} - B\|^2 + \lambda \|\tilde{X}\|_F^2 \leq (1+\varepsilon)\Delta_*$.

▶ **Theorem 18.** *There are dimensions within a constant factor of those given in Thm. 15, such that for S_1 a sparse embedding and S_2 SRHT with those dimensions, $S = S_2 S_1$ satisfies the conditions of Lemma 17, therefore the corresponding $\tilde{X}$ does as well. That is, in time*

$$O(\mathtt{nnz}(A) + \mathtt{nnz}(B)) + \tilde{O}((d + d')(\mathtt{sd}_\lambda(A)/\varepsilon + \mathtt{sd}_\lambda(A)^2)$$

time, a multiple-response ridge regression problem with n rows can be reduced to one with $\tilde{O}(\varepsilon^{-1}\, \mathtt{sd}_\lambda(A))$ rows, whose solution is a $(1+\varepsilon)$-approximate solution.

▶ Remark. Note that the solution to (15), that is, the solution to $\min_X \|\hat{S}(\hat{A}X - \hat{B})\|_F^2$, where $\hat{S}$ and $\hat{A}$ are as defined in the proof of Lemma 10, and $\hat{B} \equiv \begin{bmatrix} B \\ 0_{d \times d'} \end{bmatrix}$, is $\tilde{X} = (\hat{S}\hat{A})^+ \hat{S}\hat{B}$; that is, the matrix $\hat{A}\tilde{X} = \hat{A}(\hat{S}\hat{A})^+\hat{S}\hat{B}$ whose distance to $\hat{B}$ is within $1+\varepsilon$ of optimal has rows in the rowspace of $\hat{B}$, which is the rowspace of B. This property will be helpful building low-rank approximations.

3 Ridge Low-Rank Approximation

For an integer k we consider the problem

$$\min_{\substack{Y \in \mathbb{R}^{n \times k} \\ X \in \mathbb{R}^{k \times d}}} \|YX - A\|_F^2 + \lambda \|Y\|_F^2 + \lambda \|X\|_F^2. \tag{16}$$

From [29] (see also Corollary 43 below), this has the solution

$$\begin{aligned} Y^* &= U_k(\Sigma_k - \lambda I_k)_+^{1/2} \\ X^* &= (\Sigma_k - \lambda I_k)_+^{1/2} V_k^\top \\ \implies \mathtt{sd}_\lambda(Y^*) &= \mathtt{sd}_\lambda(X^*) = \sum_{\substack{i \in [k] \\ \sigma_i > \lambda}} (1 - \lambda/\sigma_i) \end{aligned} \tag{17}$$

where $U_k \Sigma_k V_k^\top$ is the best rank-k approximation to A, and for a matrix W, W_+ has entries that are equal to the corresponding entries of W that are nonnegative, and zero otherwise.

While [29] gives a general argument, it was also known (see for example [27]) that when the rank k is large enough not to be an active constraint (say, $k = \mathtt{rank}(A)$), then $Y^* X^*$ for Y^*, X^* from (17) solves

$$\min_{Z \in \mathbb{R}^{n \times d}} \|Z - A\|_F^2 + 2\lambda \|Z\|_*,$$

where $\|Z\|_*$ is the nuclear norm of X (also called the trace norm).

It is also well-known that

$$\|Z\|_* = \frac{1}{2}(\min_{YX = Z} \|Y\|_F^2 + \|X\|_F^2),$$

so that the optimality of (17) follows for large k.

▶ **Lemma 19.** *Given integer $k \geq 1$ and $\varepsilon > 0$, Y^* and X^* as in* (17)*, there are*

$$m = \tilde{O}(\varepsilon^{-1}\,\mathtt{sd}_\lambda(Y^*)) = \tilde{O}(\varepsilon^{-1}k) \text{ and } m' = \tilde{O}(\varepsilon^{-1}\min\{k, \varepsilon^{-1}\,\mathtt{sd}_\lambda(Y^*)\}),$$

such that there is a distribution on $S \in \mathbb{R}^{m\times n}$ and $R \in \mathbb{R}^{d\times m'}$ so that for

$$Z_S^*, Z_R^* \equiv \operatorname*{argmin}_{\substack{Z_S\in\mathbb{R}^{k\times m}\\ Z_R\in\mathbb{R}^{m'\times k}}} \|ARZ_RZ_SSA - A\|_F^2 + \lambda\|ARZ_R\|_F^2 + \lambda\|Z_SSA\|_F^2,$$

with constant probability $\tilde{Y} \equiv ARZ_R^*$ and $\tilde{X} \equiv Z_S^*SA$ satisfy

$$\|\tilde{Y}\tilde{X} - A\|_F^2 + \lambda\|\tilde{Y}\|_F^2 + \lambda\|\tilde{X}\|_F^2 \leq (1+\varepsilon)(\|Y^*X^* - A\|_F^2 + \lambda\|Y^*\|_F^2 + \lambda\|X^*\|_F^2).$$

The products SA and AR take altogether $O(\mathtt{nnz}(A)) + \tilde{O}((n+d)(\varepsilon^{-2}\,\mathtt{sd}_\lambda(Y^*) + \varepsilon^{-1}\,\mathtt{sd}_\lambda(Y^*)^2)$ to compute.

Proof. Omitted in this version. ◀

We can reduce to an even yet smaller problem, using affine embeddings, which are built using subspace embeddings. These are defined next.

▶ **Definition 20** (subspace embedding)**.** Matrix $S \in \mathbb{R}^{m_S\times n}$ is a *subspace ε-embedding* for A with respect to the Euclidean norm if $\|SAx\|_2 = (1\pm\varepsilon)\|Ax\|_2$ for all x.

▶ **Lemma 21.** *There are* sparse embedding *distributions on matrices $S \in \mathbb{R}^{m\times n}$ with $m = O(\varepsilon^{-2}\,\mathtt{rank}(A)^2)$ so that SA can be computed in $\mathtt{nnz}(A)$ time, and with constant probability S is a subspace ε-embedding. The SRHT (of Corollary 14) is a distribution on $S \in \mathbb{R}^{m\times n}$ with $m = \tilde{O}(\varepsilon^{-2}\,\mathtt{rank}(A))$ such that S is a subspace embedding with constant probability.*

Proof. The sparse embedding claim is from [10], sharpened by [24, 23]; the SRHT claim is from for example [7]. ◀

▶ **Definition 22** (Affine Embedding)**.** For A as usual and $B \in \mathbb{R}^{n\times d'}$, matrix S is an *affine ε-embedding* for A, B if $\|S(AX-B)\|_F^2 = (1\pm\varepsilon)\|AX-B\|_F^2$ for all $X \in \mathbb{R}^{d\times d'}$. A distribution over $\mathbb{R}^{m_S\times n}$ is a *poly-sized affine embedding* distribution if there is $m_S = \mathrm{poly}(d/\varepsilon)$ such that constant probability, S from the distribution is an affine ε-embedding.

▶ **Lemma 23.** *For A as usual, $B \in \mathbb{R}^{n\times d'}$, suppose there is a distribution over $S \in \mathbb{R}^{m\times n}$ so that with constant probability, S is a subspace embedding for A with parameter ε, and for $X^* \equiv \operatorname{argmin}_{X\in\mathbb{R}^{d\times d'}}\|AX - B\|_F^2$ and $B^* \equiv AX^* - B$, $\|SB*\|_F^2 = (1\pm\varepsilon)\|B^*\|_F^2$ and $\|U^\top S^\top SB^* - U^\top B^*\| \leq \varepsilon\|B^*\|_F^2$. Then S is an affine embedding for A, B. A sparse embedding with $m = O(\mathtt{rank}(A)^2/\varepsilon^2)$ has the needed properties. By first applying a sparse embedding Π, and then a Subsampled Randomized Hadamard Transform (SHRT) T, there is an affine ε-embedding $S = T\Pi$ with $m = \tilde{O}(\mathtt{rank}(A)/\varepsilon^2)$ taking time $O(\mathtt{nnz}(A) + \mathtt{nnz}(B)) + \tilde{O}((d+d')\,\mathtt{rank}(A)^{1+\kappa}/\varepsilon^2)$ time to apply to A and B, that is, to compute $SA = T\Pi A$ and SB. Here $\kappa > 0$ is any fixed value.*

Proof. Shown in [10], sharpened with [24, 23]. ◀

▶ **Theorem 24.** *With notation as in Lemma 19, there are*

$$p' = \tilde{O}(\varepsilon^{-2}m) = \tilde{O}(\varepsilon^{-3}\,\mathtt{sd}_\lambda(Y^*)) = \tilde{O}(\varepsilon^{-3}k) \text{ and}$$
$$p = \tilde{O}(\varepsilon^{-2}m') = \tilde{O}(\varepsilon^{-3}\min\{k, \varepsilon^{-1}\,\mathtt{sd}_\lambda(Y^*)\}),$$

such that there is a distribution on $S_2 \in \mathbb{R}^{p \times n}$, $R_2 \in \mathbb{R}^{d \times p'}$ *so that for*

$$\tilde{Z}_S, \tilde{Z}_R \equiv \operatorname*{argmin}_{\substack{Z_S \in \mathbb{R}^{k \times m} \\ Z_R \in \mathbb{R}^{m' \times k}}} \|S_2 ARZ_R Z_S SAR_2 - S_2 AR_2\|_F^2 + \lambda \|S_2 ARZ_R\|_F^2 + \lambda \|Z_S SAR_2\|_F^2,$$

with constant probability $\tilde{Y} \equiv AR\tilde{Z}_R$ *and* $\tilde{X} \equiv \tilde{Z}_S SA$ *satisfy*

$$\|\tilde{Y}\tilde{X} - A\|_F^2 + \lambda \|\tilde{Y}\|_F^2 + \lambda \|\tilde{X}\|_F^2 \le (1+\varepsilon)(\|Y^* X^* - A\|_F^2 + \lambda \|Y^*\|_F^2 + \lambda \|X^*\|_F^2).$$

The matrices $S_2 AR$, SAR, *and* SAR_2 *can be computed in* $O(\mathtt{nnz}(A)) + \operatorname{poly}(\mathtt{sd}_\lambda(Y^*)/\varepsilon)$ *time.*

Proof. Omitted in this version. ◀

▶ **Lemma 25.** *For* $C \in \mathbb{R}^{p \times m'}$, $D \in \mathbb{R}^{m \times p'}$, $G \in \mathbb{R}^{p \times p'}$, *the problem of finding*

$$\min_{\substack{Z_S \in \mathbb{R}^{k \times m} \\ Z_R \in \mathbb{R}^{m' \times k}}} \|CZ_R Z_S D - G\|_F^2 + \lambda \|CZ_R\|_F^2 + \lambda \|Z_S D\|_F^2, \tag{18}$$

and the minimizing CZ_R *and* $Z_S D$, *can be solved in*

$$O(pm'r_C + p'mr_D + r_D p(p' + r_C))$$

time, where $r_C \equiv \mathtt{rank}(C) \le \min\{m', p\}$, *and* $r_D \equiv \mathtt{rank}(D) \le \min\{m, p'\}$.

Proof. Please see §E. ◀

▶ **Theorem 26.** *The matrices* $\tilde{Z}_S, \tilde{Z}_R$ *of Theorem 24 can be found in*

$$O(\mathtt{nnz}(A)) + \operatorname{poly}(\mathtt{sd}_\lambda(Y^*)/\varepsilon)$$

time, in particular $O(\mathtt{nnz}(A)) + \tilde{O}(\varepsilon^{-7}\,\mathtt{sd}_\lambda(Y^*)^2\ \min\{k, \varepsilon^{-1}\,\mathtt{sd}_\lambda(Y^*)\})$ *time, such that with constant probability,* $AR\tilde{Z}_R, \tilde{Z}_S SA$ *is an* ε*-approximate minimizer to* (16), *that is,*

$$\begin{aligned} &\|(AR\tilde{Z}_R)(\tilde{Z}_S SA) - A\|_F^2 + \lambda \|AR\tilde{Z}_R\|_F^2 + \lambda \|\tilde{Z}_S SA\|_F^2 && (19)\\ &\qquad \le (1+\varepsilon) \min_{\substack{Y \in \mathbb{R}^{n \times k} \\ X \in \mathbb{R}^{k \times d}}} \|YX - A\|_F^2 + \lambda \|Y\|_F^2 + \lambda \|X\|_F^2. && (20) \end{aligned}$$

With an additional $O(n+d)\operatorname{poly}(\mathtt{sd}_\lambda(Y^*)/\varepsilon)$ *time, and in particular*

$$\tilde{O}(\varepsilon^{-1} k\, \mathtt{sd}_\lambda(Y^*)(n + d + \min\{n, d\} \min\{k/\,\mathtt{sd}_\lambda(Y^*), \varepsilon^{-1}\}))$$

time, the solution matrices $\tilde{Y} \equiv AR\tilde{Z}_R, \tilde{X} \equiv \tilde{Z}_S SA$ *can be computed and output.*

An expression for $\mathtt{sd}_\lambda(Y^*)$ is given at (17).

Proof. Follows from Theorem 24 and Lemma 25, noting that for efficiency's sake we can use the transpose of A instead of A. ◀

Acknowledgement. We acknowledge the support from XDATA program of the Defense Advanced Research Projects Agency (DARPA), administered through Air Force Research Laboratory contract FA8750-12-C-0323.

References

1 Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *ACM Symposium on Theory of Computing (STOC)*, 2006.

2 Alexandr Andoni and Huy L. Nguyen. Eigenvalues of a matrix in the streaming model. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1729–1737. Society for Industrial and Applied Mathematics, 2013.

3 Haim Avron, Christos Boutsidis, Sivan Toledo, and Anastasios Zouzias. Efficient dimensionality reduction for canonical correlation analysis. *SIAM Journal on Scientific Computing*, 36(5):S111–S131, 2014. doi:10.1137/130919222.

4 Haim Avron, Kenneth L. Clarkson, and David P. Woodruff. Faster kernel ridge regression using sketching and preconditioning. *CoRR*, abs/1611.03220, 2016. URL: http://arxiv.org/abs/1611.03220.

5 A. Björck and G. H. Golub. Numerical methods for computing angles between linear subspaces. *Mathematics of Computation*, 27(123):579–594, 1973.

6 Jean Bourgain, Sjoerd Dirksen, and Jelani Nelson. Toward a unified theory of sparse dimensionality reduction in euclidean space. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 499–508, 2015.

7 C. Boutsidis and A. Gittens. Improved matrix algorithms via the Subsampled Randomized Hadamard Transform. *ArXiv e-prints*, March 2012. arXiv:1204.0062.

8 Ricardo Cabral, Fernando De la Torre, João P Costeira, and Alexandre Bernardino. Unifying nuclear norm and bilinear factorization approaches for low-rank matrix decomposition. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2488–2495. IEEE, 2013.

9 Shouyuan Chen, Yang Liu, Michael Lyu, Irwin King, and Shengyu Zhang. Fast relative-error approximation algorithm for ridge regression. In *31st Conference on Uncertainty in Artificial Intelligence*, 2015.

10 Kenneth L. Clarkson and David P. Woodruff. Low rank approximation and regression in input sparsity time. In *STOC*, 2013. Full version at http://arxiv.org/abs/1207.6365.

11 M. B. Cohen, S. Elder, C. Musco, C. Musco, and M. Persu. Dimensionality Reduction for k-Means Clustering and Low Rank Approximation. *ArXiv e-prints*, October 2014. arXiv:1410.6801.

12 Michael B. Cohen. Nearly tight oblivious subspace embeddings by trace inequalities. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 278–287, 2016.

13 Michael B. Cohen, Jelani Nelson, and David P. Woodruff. Optimal approximate matrix product in terms of stable rank. *CoRR*, abs/1507.02268, 2015.

14 P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Sampling algorithms for ℓ_2 regression and applications. In *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1127–1136, 2006.

15 P. Drineas, M. W. Mahoney, S. Muthukrishnan, and T. Sarlos. Faster least squares approximation, Technical Report, arXiv:0710.1435, 2007. URL: http://www.citebase.org/abstract?id=oai:arXiv.org:0710.1435.

16 Petros Drineas, Michael W. Mahoney, Malik Magdon-Ismail, and David P. Woodruff. Fast approximation of matrix coherence and statistical leverage. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 – July 1, 2012*, 2012.

17 Ahmed El Alaoui and Michael W. Mahoney. Fast randomized kernel methods with statistical guarantees. *stat*, 1050:2, 2014.

18 R. Frostig, R. Ge, S. M. Kakade, and A. Sidford. Competing with the Empirical Risk Minimizer in a Single Pass. *ArXiv e-prints*, December 2014. Appeared in COLT 2015. `arXiv:1412.6606`.

19 R. Frostig, R. Ge, S. M. Kakade, and A. Sidford. Un-regularizing: approximate proximal point and faster stochastic algorithms for empirical risk minimization. In *International Conference on Machine Learning (ICML)*, 2015.

20 Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2013.

21 Yichao Lu, Paramveer Dhillon, Dean P Foster, and Lyle Ungar. Faster ridge regression via the subsampled randomized hadamard transform. In *Advances in Neural Information Processing Systems*, pages 369–377, 2013.

22 Michael W. Mahoney. Randomized algorithms for matrices and data. *Found. Trends Mach. Learn.*, 3(2):123–224, February 2011. `doi:10.1561/2200000035`.

23 Xiangrui Meng and Michael W. Mahoney. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *STOC*, pages 91–100, 2013.

24 Jelani Nelson and Huy L. Nguyen. OSNAP: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *FOCS*, pages 117–126, 2013.

25 Mert Pilanci and Martin J. Wainwright. Randomized sketches of convex programs with sharp guarantees. *CoRR*, abs/1404.7203, 2014. URL: `http://arxiv.org/abs/1404.7203`.

26 T. Sarlós. Improved approximation algorithms for large matrices via random projections. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2006.

27 Nathan Srebro and Adi Shraibman. Rank, trace-norm and max-norm. In *Learning Theory*, pages 545–560. Springer, 2005.

28 Joel Tropp. Improved analysis of the subsampled randomized Hadamard transform. *Adv. Adapt. Data Anal., Special Issue, "Sparse Representation of Data and Images"*, 2011.

29 M. Udell, C. Horn, R. Zadeh, and S. Boyd. Generalized Low Rank Models. *ArXiv e-prints*, October 2014. `arXiv:1410.0342`.

30 David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014. `doi:10.1561/0400000060`.

31 Jiyan Yang, Xiangrui Meng, and M. W. Mahoney. Implementing randomized matrix algorithms in parallel and distributed environments. *Proceedings of the IEEE*, 104(1):58–92, Jan 2016. `doi:10.1109/JPROC.2015.2494219`.

32 Y. Yang, M. Pilanci, and M. J. Wainwright. Randomized sketches for kernels: Fast and optimal non-parametric regression. *ArXiv e-prints*, January 2015. `arXiv:1501.06195`.

33 Dean Foster Yichao Lu, Paramveer Dhillon and Lyle Ungar. Faster ridge regression via the subsampled randomized hadamard transform. In *Proceedings of the Neural Information Processing Systems (NIPS) Conference*, 2013.

A Estimation of statistical dimension

▶ **Theorem 27.** *If the statistical dimension* $\mathtt{sd}_\lambda(A)$ *is at most*

$$M \equiv \min\{n, d, \lfloor (n+d)^{1/3}/\mathrm{poly}(\log(n+d))\rfloor\},$$

it can be estimated to within a constant factor in $O(\mathtt{nnz}(A))$ *time, with constant probability.*

Proof. From Lemma 18 of [11], generalizing the machinery of [2], the first z squared singular values of A can be estimated up to additive $\frac{\varepsilon}{z}\|A_{-z}\|_F^2$ in time $O(\mathtt{nnz}(A)) + \tilde{O}(z^3/\mathrm{poly}(\varepsilon))$, where $A_{-z} \equiv A - A_z$ denotes the residual error of the best rank-z approximation A_z to A. Therefore $\|A_z\|_F^2$ can be estimated up to additive $\varepsilon\|A_{-z}\|_F^2$, and the same for $\|A_{-z}\|_F^2$. This implies that for small enough constant ε, $\|A_{-z}\|_F^2$ can be estimated up to constant relative error, using the same procedure.

Thus in $O(\mathtt{nnz}(A))$ time, the first $6M$ singular values of A can be estimated up to additive $\frac{1}{6M}\|A_{-6M}\|_F^2$ error, and there is an estimator $\hat{\gamma}_z$ of $\|A_{-z}\|_F^2$ up to relative error $1/3$, for $z \in [6M]$.

Since $1/(1+\lambda/\sigma_i^2) \leq \min\{1, \sigma_i^2/\lambda\}$, for any z the summands of $\mathtt{sd}_\lambda(A)$ for $i \leq z$ are at most 1, while those for $i > z$ are at most σ_i^2/λ, and so $\mathtt{sd}_\lambda(A) \leq z + \|A_{-z}\|_F^2/\lambda$.

When $\sigma_z^2 \leq \lambda$, the summands of $\mathtt{sd}_\lambda(A)$ for $i \geq z$ are at least $\frac{1}{2}\frac{\sigma_i^2}{\lambda}$, and so $\mathtt{sd}_\lambda(A) \geq \frac{1}{2}\|A_{-z}\|_F^2/\lambda$. When $\sigma_z^2 \geq \lambda$, the summands of $\mathtt{sd}_\lambda(A)$ for $i \leq z$ are at least $1/2$. Therefore $\mathtt{sd}_\lambda(A) \geq \frac{1}{2}\min\{z, \|A_{-z}\|_F^2/\lambda\}$.

Under the constant-probability assumption that $\hat{\gamma}_z = (1 \pm 1/3)\|A_{-z}\|_F^2$, we have

$$\frac{3}{8}\min\{z, \hat{\gamma}_z/\lambda\} \leq \mathtt{sd}_\lambda(A) \leq \frac{3}{2}(z + \hat{\gamma}_z/\lambda). \tag{21}$$

Let z' be the smallest z of the form 2^j for $j = 0, 1, 2, \ldots$, with $z' \leq 6M$, such that $z' \geq \hat{\gamma}_{z'}/\lambda$. Since $M \geq \mathtt{sd}_\lambda(A) \geq \frac{3}{8}z$ for $z \leq \hat{\gamma}_z/\lambda$, there must be such a z'. Then by considering the lower bound of (21) for z' and for $z'/2$, we have $\mathtt{sd}_\lambda(A) \geq \frac{3}{8}\max\{z'/2, \hat{\gamma}_{z'}/\lambda\} \geq \frac{1}{16}(z' + \hat{\gamma}_{z'}/\lambda)$, which combined with the upper bound of (21) implies that $z' + \hat{\gamma}_{z'}/\lambda$ is an estimator of $\mathtt{sd}_\lambda(A)$ up to a constant factor. ◀

B Regularized Canonical Correlation Analysis

First, we show how to compute regularized CCA using a modified Björck-Golub algorithm.

▶ **Definition 28.** Let $A \in \mathbb{R}^{n \times d}$ with $n \geq d$ and let $\lambda \geq 0$. $A = QR$ is a *λ-QR factorization* if Q is full rank, R is upper triangular and $R^\top R = A^\top A + \lambda I_d$.

▶ Remark. A λ-QR factorization always exists, and R will be invertible for $\lambda > 0$. Q has orthonormal columns for $\lambda = 0$.

▶ **Fact 29.** *For a λ-QR factorization $A = QR$ we have $Q^\top Q + \lambda R^{-\top}R^{-1} = I_d$.*

Proof. A direct consequence of $R^\top R = A^\top A + \lambda I_d$ (multiply from the right by R^{-1} and the left by $R^{-\top}$). ◀

▶ **Fact 30.** *For a λ-QR factorization $A = QR$ we have $\mathtt{sd}_\lambda(A) = \|Q\|_F^2$.*

Proof. Omitted in this version. ◀

▶ **Theorem 31** (Regularized Björck-Golub). *Let $A = Q_A R_A$ be a λ_1-QR factorization of A, and $B = Q_B R_B$ be a λ_2-QR factorization of B. Assume that $\lambda_1 > 0$ and $\lambda_2 > 0$. The (λ_1, λ_2) canonical correlations are exactly the singular values of $Q_A^\top Q_B$. Furthermore, if $Q_A^\top Q_B = M\Sigma N^T$ is a thin SVD of $Q_A^\top Q_B$, then the columns of $R_A^{-1}M$ and $R_B^{-1}N$ are canonical weights.*

Proof. Omitted in this version. ◀

We now consider how to approximate the computation using sketching. The basic idea is similar to the one used in [3] to accelerate the computation of non-regularized CCA: compute the regularized canonical correlations and canonical weights of the pair (SA, SB) for a sufficiently large subspace embedding matrix S. Similarly to [3], we define the notion of approximate regularized CCA, and show that for large enough S we find an approximate CCA with high probability.

▶ **Definition 32** (Approximate (λ_1, λ_2) regularized CCA)**.** For $0 \le \eta \le 1$, an η-approximate (λ_1, λ_2) regularized CCA of (A, B) is a set of positive numbers $\hat{\sigma}_1 \ge \cdots \ge \hat{\sigma}_q$, and vectors $\hat{u}_1, \ldots, \hat{u}_q \in \mathbb{R}^d$ and $\hat{v}_1, \ldots, \hat{v}_q \in \mathbb{R}^{d'}$ such that

(a) For every i,

$$\left|\hat{\sigma}_i - \sigma_i^{(\lambda_1,\lambda_2)}\right| \le \eta\,.$$

(b) Let $\hat{U} = [\hat{u}_1, \ldots, \hat{u}_q] \in \mathbb{R}^{n \times q}$ and $\hat{V} = [\hat{v}_1, \ldots, \hat{v}_q] \in \mathbb{R}^{d' \times q}$. We have,

$$\left|\hat{U}^\top(A^\top A + \lambda_1 I_d)\hat{U} - I_q\right| \le \eta$$

and

$$\left|\hat{V}^\top(B^\top B + \lambda_2 I_{d'})\hat{V}s - I_q\right| \le \eta\,.$$

In the above, the notation $|X| \le \alpha$ should be understood as entry-wise inequality.

(c) For every i,

$$\left|\hat{u}_i^\top A^\top B \hat{v}_i - \sigma_i^{(\lambda_1,\lambda_2)}\right| \le \eta\,.$$

▶ **Theorem 33.** *If S is a sparse embedding matrix with $m = \Omega(\max(\mathtt{sd}_{\lambda_1}(A), \mathtt{sd}_{\lambda_2}(B))^2/\epsilon^2)$ rows, then with high probability the (λ_1, λ_2) canonical correlations and canonical weights of (SA, SB) form an ϵ-approximate (λ_1, λ_2) regularized CCA for (A, B).*

Proof. Omitted in this version. ◀

Taking an optimization point of view, the following Corollary shows that the suboptimality in the objective is not too big (the fact that the constraints are approximately held is established in the previous theorem).

▶ **Corollary 34.** *Let U_L and V_L (respectively, $\hat{U}_L$ and $\hat{V}_L$) denote the first L columns of U and V (respectively, $\hat{U}$ and $\hat{V}$. Then,*

$$\mathtt{tr}(\hat{U}_L^\top A^\top B \hat{V}_L) \le \mathtt{tr}(U_L^\top A^\top B V_L) + \epsilon L\,.$$

C General Regularization: Multiple-response Regression

In this section we consider the problem

$$X^* \equiv \operatorname*{argmin}_{X \in \mathbb{R}^{d \times d'}} \|AX - B\|_F^2 + f(X)$$

for a real-valued function f on matrices. We show that under certain assumptions on f (generalizing from $f(X) = \|X\|_h$ for some orthogonally invariant norm $\|\cdot\|_h$), if we have an approximation algorithm for the problem, then via sketching the running time dependence of the algorithm on n can be improved.

▶ **Definition 35** ((left/right) orthogonal invariance($\mathtt{loi}$/$\mathtt{roi}$))**.** A matrix measure $f()$ is *left orthogonally invariant* (or $\mathtt{loi}$ for short) if $f(UA) = f(A)$ for all A and orthogonal U. Similarly define *right orthogonal invariance* ($\mathtt{roi}$). Note that $f()$ is orthogonally invariant if it is both left and right orthogonally invariant.

When norm $\|\cdot\|_g$ is orthogonally invariant, it can be expressed as $\|A\|_g = g(\sigma_1, \sigma_2, \ldots, \sigma_r)$, where the σ_i are the singular values of A, and $g()$ is a *symmetric gauge function*: a function that is even in each argument, and symmetric, meaning that its value depends only on the set of input values and not their order.

▶ **Definition 36** (padding invariance). Say that a matrix measure $f()$ is *padding invariant* if it is preserved by padding A with rows or columns of zeroes: $f(\left[\begin{smallmatrix} A \\ 0_{z\times d}\end{smallmatrix}\right]) = f\left(\begin{smallmatrix} A & 0_{n\times z'}\end{smallmatrix}\right) = f(A)$.

▶ **Lemma 37.** *Unitarily invariant norms and v-norms are padding invariant.*

Proof. Omitted in this version. ◀

▶ **Definition 38** (`piloi`, `piroi`). Say that a matrix measure is `piloi` if it is padding invariant and left orthogonally invariant, and `piroi` if it is padding invariant and right orthogonally invariant.

The following is the main theorem of this section.

▶ **Theorem 39.** *Let $f()$ be a real-valued function on matrices that is* `piroi` *and subadditive. Let $B \in \mathbb{R}^{n\times d'}$. Let*

$$X^* \equiv \operatorname*{argmin}_{X\in\mathbb{R}^{d\times d'}} \|AX - B\|_F^2 + f(X), \tag{22}$$

and $\Delta_* \equiv \|AX^* - B\|_F^2 + f(X^*)$. Suppose that for $r \equiv \texttt{rank}\, A$, there is an algorithm that for general n, d, d', r and $\varepsilon > 0$, finds $\tilde{X}$ with $\|A\tilde{X} - B\|_F^2 + f(\tilde{X}) \leq (1+\varepsilon)\Delta_*$ in time $\tau(d, n, d', r, \varepsilon)$. Then there is an algorithm that with constant probability finds such a $\tilde{X}$, taking time

$$O(\texttt{nnz}(A) + \texttt{nnz}(B) + (n + d + d')\text{poly}(r/\varepsilon)) + \tau(d, \text{poly}(r/\varepsilon), \text{poly}(r/\varepsilon), r, \varepsilon).$$

Although earlier results for constrained least squares (e.g. [10]) can be applied to obtain approximation algorithms for regularized multiple-response least squares, via the solution of $\min_{X\in\mathbb{R}^{d\times d'}} \|AX - B\|_F^2$, subject to $f(X) \leq C$ for a chosen constant C, such a reduction yields a slower algorithm if properties of $f(X)$ are not exploited, as here.

Proof. Omitted in this version. ◀

D General Regularization: Low-rank Approximation

For an integer k we consider the problem

$$\min_{\substack{Y\in\mathbb{R}^{n\times k}\\ X\in\mathbb{R}^{k\times d}}} \|YX - A\|_F^2 + f(Y, X), \tag{23}$$

where $f(\cdot,\cdot)$ is a real-valued function that is `piloi` in the left argument, `piroi` in the right argument, and left and right reduced by contraction in its left and right arguments, respectively.

For example $\hat{f}(\|Y\|_\ell, \|X\|_r)$ for `piloi` $\|\cdot\|_\ell$ and `piroi` $\|\cdot\|_r$ would satisfy these conditions, as would $\|YX\|_g$ for orthogonally invariant norm $\|\cdot\|_g$. The function $\hat{f}$ could be zero for arguments whose maximum is less than some μ, and infinity otherwise.

D.1 Via the SVD

First, a solution method relying on the singular value decomposition for a slightly more general problem than (23).

▶ **Theorem 40.** *Let k be a positive integer, $f_1 : \mathbb{R} \mapsto \mathbb{R}$ increasing, and $f : \mathbb{R}^{n\times k} \times \mathbb{R}^{k\times d} \mapsto \mathbb{R}$, where f is* `piloi` *and subadditive in its left argument, and* `piroi` *and subadditive in in its right argument.*

Let A have full SVD $A = U\Sigma V^\top$, $\Sigma_k \in \mathbb{R}^{k\times k}$ the diagonal matrix of top k singular values of A. Let matrices $W^*, Z^* \in \mathbb{R}^{k\times k}$ solve

$$\min_{\substack{W\in\mathbb{R}^{k\times k}\\ Z\in\mathbb{R}^{k\times k}\\ WZ \text{ diagonal}}} \quad f_1(\|WZ - \Sigma_k\|_{(p)}) + f(W, Z), \tag{24}$$

and suppose there is a procedure taking $\tau(k)$ time to find W^* and Z^*. Then the solution to

$$\min_{\substack{Y\in\mathbb{R}^{n\times k}\\ X\in\mathbb{R}^{k\times d}}} f_1(\|YX - A\|_{(p)}) + f(Y, X) \tag{25}$$

is $Y^* = U\begin{bmatrix} W^* \\ 0_{(n-k)\times k}\end{bmatrix}$ and $X^* = \begin{bmatrix} Z^* & 0_{k\times(d-k)}\end{bmatrix} V^\top$. Thus for general A, (25) can be solved in time $O(nd\min\{n, d\}) + \tau(k)$.

Proof. Omitted in this version. ◀

We sharpen this result for the case that the regularization term comes from orthogonally invariant norms.

▶ **Theorem 41.** *Consider (25) when $f(\cdot,\cdot)$ has the form $\hat{f}(\|Y\|_\ell, \|X\|_r)$, where $\|\cdot\|_\ell$ and $\|\cdot\|_r$ are orthogonally invariant, and $\hat{f} : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$ increasing in each argument. Suppose in that setting there is a procedure that solves (25) when A, Y, and X are diagonal matrices, taking time $\tau(r)$ for a function $\tau(\cdot)$, with $r \equiv \texttt{rank}(A)$. Then for general A, (25) can be solved by finding the SVD of A, and applying the given procedure to $k \times k$ diagonal matrices, taking altogether time $O(nd\min\{n, d\}) + \tau(k)$.*

Proof. Omitted in this version. ◀

▶ **Definition 42** (clipping to nonnegative $(\cdot)_+$)**.** For real number a, let $(a)_+$ denote a, if $a \geq 0$, and zero otherwise. For matrix A, let $(A)_+$ denote coordinatewise application.

▶ **Corollary 43.** *If the objective function in (25) is $\|YX - A\|_F^2 + 2\lambda\|YX\|_{(1)}$ or $\|YX - A\|_F^2 + \lambda(\|Y\|_F^2 + \|X\|_F^2)$, then the diagonal matrices W^* and Z^* from Theorem 41 yielding the solution are $W^* = Z^* = \sqrt{(\Sigma_k - \lambda I_k)_+}$, where Σ_k is the $k \times k$ diagonal matrix of top k singular values of A [29].*

If the objective function is $\|YX - A\|_{(p)} + \lambda\|YX\|_{(1)}$ for $p \in [1, \infty]$, then $W^* = Z^* = \sqrt{(\Sigma_k - \alpha I_k)_+}$, for an appropriate value α.

If the objective function is $\|YX - A\|_F^2 + \lambda\|YX\|_F^2$, then $W^* = Z^* = \sqrt{\Sigma_k/(1 + \lambda)}$.

Proof. Omitted in this version. ◀

D.2 Reduction to a small problem via sketching

▶ **Theorem 44.** *Suppose there is a procedure that solves* (23) *when A, Y, and X are $k \times k$ matrices, and A is diagonal, and YX is constrained to be diagonal, taking time $\tau(k)$ for a function $\tau(\cdot)$. Let f also inherit a sketching distribution on the left in its left argument, and on the right in its right argument. Then for general A, there is an algorithm that finds ε-approximate solution $(\tilde{Y}, \tilde{X})$ in time*

$$O(\texttt{nnz}(A)) + \tilde{O}(n + d)\mathrm{poly}(k/\varepsilon) + \tau(k).$$

Proof. Omitted in this version. ◀

E Proof of Lemma 25

Proof. Let U_C be an orthogonal basis for $\texttt{colspace}(C)$, so that every matrix of the form CZ_R is equal to $U_C Z'_R$ for some Z'_R. Similarly let $U_D^\top$ be an orthogonal basis for $\texttt{rowspan}(D)$, so that every matrix of the form $Z_S D$ is equal to one of the form $Z'_S U_D$. Let $P_C \equiv U_C U_C^\top$ and $P_D \equiv U_D U_D^\top$. Then using $P_C(I - P_C) = 0$, $P_D(I - P_D) = 0$, and matrix Pythagoras,

$$\begin{aligned}
&\|CZ_R Z_S D - G\|_F^2 + \lambda\|CZ_R\|_F^2 + \lambda\|Z_S D\|_F^2 \\
&\qquad = \|P_C U_C Z'_R Z'_S U_D^\top P_D - G\|_F^2 + \lambda\|U_C Z'_R\|_F^2 + \lambda\|Z'_S U_D^\top\|_F^2 \\
&\qquad = \|P_C U_C Z'_R Z'_S U_D^\top P_D - P_C G P_D\|_F^2 + \|(I - P_C)G\|_F^2 \\
&\qquad\qquad + \|P_C G(I - P_D)\|_F^2 + \lambda\|Z'_R\|_F^2 + \lambda\|Z'_S\|_F^2.
\end{aligned}$$

So minimizing (18) is equivalent to minimizing

$$\begin{aligned}
&\|P_C U_C Z'_R Z'_S U_D^\top P_D - P_C G P_D\|_F^2 + \lambda\|Z'_R\|_F^2 + \lambda\|Z'_S\|_F^2 \\
&\qquad = \|U_C Z'_R Z'_S U_D^\top - U_C U_C^\top G U_D U_D^\top\|_F^2 + \lambda\|Z'_R\|_F^2 + \lambda\|Z'_S\|_F^2 \\
&\qquad = \|Z'_R Z'_S - U_C^\top G U_D\|_F^2 + \lambda\|Z'_R\|_F^2 + \lambda\|Z'_S\|_F^2.
\end{aligned}$$

This has the form of (16), mapping Y of (16) to Z'_R, X to Z'_S, and A to $U_C^\top G U_D$, from which a solution of the form (17) can be obtained.

To recover Z_R from Z'_R: we have $C = U_C \left[\, T_C \; T'_C \,\right]$, for matrices T_C and T'_C, where upper triangular $T_C \in \mathbb{R}^{r_C \times r_C}$. We recover Z_R as $\begin{bmatrix} T_C^{-1} \hat{Z}'_R \\ 0_{m - r_C \times k} \end{bmatrix}$, since then $U_C Z'_R = CZ_R$. A similar back-substitution allows recovery of Z_S from Z'_S.

Running times: to compute U_C and U_D, $O(pm' r_C + mp' r_D)$; to compute $U_C^\top G U_D$, $O(r_D p(p' + r_C))$; to compute and use the SVD of $U_C^\top G U_D$ to to solve (16) via (17), $O(r_C r_D \min\{r_C, r_D\})$; to recover Z_R and Z_S, $O(k(r_C^2 + r_D^2))$. Thus, assuming $k \leq \min\{p, p'\}$ and using $r_C \leq \min\{p, m'\}$ and $r_D \leq \min\{m, p'\}$, the total running time is $O(pm' r_C + p' m r_D + pp'(r_C + r_D))$, as claimed. ◀

The Lovász Theta Function for Random Regular Graphs and Community Detection in the Hard Regime

Jess Banks[1], **Robert Kleinberg**[2], **and Cristopher Moore**[3]

1 **Department of Mathematics, University of California, Berkeley, CA, USA**
jess.m.banks@berkeley.edu

2 **Santa Fe Institute, Santa Fe, NM, USA**
moore@santafe.edu

3 **Department of Computer Science, Cornell University, Ithaca, NY, USA**
rdk@cs.cornell.edu

Abstract

We derive upper and lower bounds on the degree d for which the Lovász ϑ function, or equivalently sum-of-squares proofs with degree two, can refute the existence of a k-coloring in random regular graphs $G_{n,d}$. We show that this type of refutation fails well above the k-colorability transition, and in particular everywhere below the Kesten-Stigum threshold. This is consistent with the conjecture that refuting k-colorability, or distinguishing $G_{n,d}$ from the planted coloring model, is hard in this region. Our results also apply to the disassortative case of the stochastic block model, adding evidence to the conjecture that there is a regime where community detection is computationally hard even though it is information-theoretically possible. Using orthogonal polynomials, we also provide explicit upper bounds on $\vartheta(\overline{G})$ for regular graphs of a given girth, which may be of independent interest.

1998 ACM Subject Classification G.2.2 Graph Theory

Keywords and phrases Lovász Theta Function, Random Regular Graphs, Sum of Squares, Orthogonal Polynomials

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.28

1 Introduction

Many constraint satisfaction problems have *phase transitions* in the random case: as the ratio between the number of constraints and the number of variables increases, there is a critical value at which the probability that a solution exists, in the limit $n \to \infty$, suddenly drops from one to zero. Above this transition, most instances are too constrained and hence unsatisfiable. But how many constraints do we need before it becomes easy to *prove* that a typical instance is unsatisfiable? When is there likely to be a short refutation, which we can find in polynomial time, proving that no solution exists?

For a closely related problem, suppose that a constraint satisfaction problem is generated randomly, but with a particular solution "planted" in it. Given the instance, can we recover the planted solution, at least approximately? For that matter, can we tell whether the instance was generated from this planted model, as opposed to an un-planted model with no built-in solution? We can think of this as a statistical inference problem. If there is an underlying pattern in a dataset (the planted solution) but also some noise (the probabilistic process by which the instance is generated) the question is how much data (how many constraints) we need before we can find the pattern, or confirm that one exists.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 28; pp. 28:1–28:22
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Here we focus on the k-colorability of random graphs, and more generally the community detection problem. Let $G = G(n, p = d/n)$ denote the Erdős-Rényi graph with n vertices and average degree d. A simple first moment argument shows that with high probability G is is not k-colorable if

$$d \geq d_{\text{first}} = 2k \ln k - \ln k \,. \tag{1}$$

(We say that an event E_n on graphs of size n holds with high probability if $\lim_{n\to\infty} \Pr[E_n] = 1$, and with positive probability if $\liminf_{n\to\infty} \Pr[E_n] > 0$.) Sophisticated uses of the second moment method [8, 24] shows that this is essentially tight, and that the k-colorability transition occurs at

$$d_c = d_{\text{first}} - O(1) \,.$$

Now consider the planted coloring model, where we choose a coloring σ uniformly at random and condition G on the event that σ is proper.

If $d > d_c$, then $G(n, d/n)$ is probably not k-colorable, while graphs drawn from the planted model are k-colorable by construction. Thus, above the k-colorability transition, we can tell with high probability whether G was drawn from the planted or un-planted model by checking to see if G is k-colorable. However, searching exhaustively for k-colorings would take exponential time.

A similar situation holds for the stochastic block model, a model of graphs with community structure also known as the planted partition problem (see [47, 2] for reviews). For our purposes, we will define it as follows: fix a constant τ, and say a partition σ of the vertices into k groups is "good" if a fraction τ/k of the edges connect vertices within groups. Equivalently, if G has m edges, σ is a multiway cut with $(1 - \tau/k)m$ edges crossing between groups. Generalizing the planted coloring model where $\tau = 0$, the block model chooses σ uniformly, and conditions G on the event that σ is good. The cases $\tau > 1$ and $\tau < 1$, where vertices are more or less likely to be connected to others in the same group, are called *assortative* (or *ferromagnetic*) and *disassortative* (or *antiferromagnetic*) respectively.

Two natural problems related to the block model are *detection*, i.e., telling with high probability whether G was drawn from the block model or from $G(n, d/n)$, and *reconstruction*, finding a partition which is significantly correlated with the planted partition σ. (This is sometimes called *weak* reconstruction to distinguish it from finding σ exactly, which becomes possible when $d = \Theta(\log n)$ [16, 1, 3, 30, 31, 9, 50].) Both problems become information-theoretically possible at a point called the condensation transition [39, 22, 19], and the first and second moment methods [12] show that this scales as

$$d_c \sim \frac{k \log k}{(\tau - 1)^2} \,, \tag{2}$$

where $\sim$ hides a multiplicative constant. As in k-coloring this is roughly the first-moment bound above which, with high probability, no good partitions exist in $G(n, d/n)$. However, the obvious algorithms for detection and reconstruction, such as searching exhaustively for good partitions or sampling from an appropriate Gibbs distribution [6, 4], require exponential time.

In fact, conjectures from statistical physics [40, 25, 26] suggest this exponential difficulty is sometimes unavoidable. Specifically, these conjectures state that polynomial-time algorithms for detection and reconstruction exist if and only if d is above the *Kesten-Stigum threshold* [34, 35],

$$d_{\text{KS}} = \left(\frac{k-1}{\tau - 1} \right)^2 \,. \tag{3}$$

Several polynomial-time algorithms are now known to succeed whenever $d > d_{\mathrm{KS}}$, including variants of belief propagation [49, 5] and spectral algorithms based on non-backtracking walks [48, 38, 43, 17]. Moreover, for $k = 2$ we know that the information-theoretic and Kesten-Stigum thresholds coincide [51]. Comparing (2) and (3) we see that for any $\tau \neq 1$ we have $d_c < d_{\mathrm{KS}}$ for sufficiently large k, and in fact this occurs for some $\tau < 1$ when $k = 4$ and more generally when $k \geq 5$ [6, 4, 12].

Thus in the regime $d_c < d < d_{\mathrm{KS}}$, detection and reconstruction are information-theoretically possible, but are conjectured to be computationally hard. In particular, this conjecture implies that there is no way to refute the existence of a coloring, or of a good partition, whenever $d < d_{\mathrm{KS}}$, even when d is large enough so that a coloring or partition probably does not exist. Our goal in this paper is to rule out spectral refutations based on the Lovász theta function, or equivalently sum-of-squares proofs of degree two.

For technical reasons, we focus on random d-regular graphs, which we denote $G_{n,d}$. A series of papers applying the first and second moment methods in this setting [46, 7, 33, 21] have determined the likely chromatic number of $G_{n,d}$ for almost all d, showing that the critical d for k-colorability is $d_c = d_{\mathrm{first}} - O(1)$ just as for $G(n, d/n)$. (There are a few values of d and k where $G_{n,d}$ could be k-colorable with probability strictly between 0 and 1, so this transition might not be completely sharp.)

We define the d-regular block model by choosing a planted partition σ uniformly at random and conditioning $G_{n,d}$ on the event that σ is good. Equivalently, we choose G uniformly from all d-regular graphs such that a fraction τ/k of their $m = dn/2$ edges connect vertices within groups. We claim that our results also apply to the regular block model proposed in [51] where d-regular graphs are chosen with probability proportional to $\tau^{\#\text{ within-group edges}}((k-\tau)/(k-1))^{\#\text{ between-group edges}}$: in that case, the fraction of within-group edges fluctuates, but is $\tau/k + o(1)$ with high probability.[1] We again conjecture that refuting the existence of a coloring or a good partition is exponentially hard below the Kesten-Stigum bound. Since the branching ratio of a d-regular tree is $d - 1$, in the regular case this becomes

$$d < d_{\mathrm{KS}} = \left(\frac{k-1}{\tau-1}\right)^2 + 1\,.$$

Main Results

The Lovász ϑ function, which we review below, gives a lower bound on the chromatic number which can be computed in polynomial time. In particular, if $\vartheta(\overline{G}) > k$, this provides a polynomial-time refutation of G's k-colorability. We first prove that this type of refutation exactly corresponds to sum-of-squares proofs of degree two in a natural encoding of k-colorability as a system of polynomials; the connection between SDP relaxations and degree-two SOS is standard [54] but we give an explicit proof here for completeness. We then show the following bounds on the likely value of $\vartheta(\overline{G})$ when G is a random d-regular graph.

▶ **Theorem 1.** *Let d be constant. For any constant $\epsilon > 0$, with high probability*

$$\frac{d}{2\sqrt{d-1}} + 1 - \epsilon \leq \vartheta(\overline{G_{n,d}}) \leq \frac{d}{2\sqrt{d-1}} + 2 + \epsilon\,. \tag{4}$$

[1] These models are not to be confused with a stricter model, where for some constants q_{rs} each vertex in group r has exactly q_{rs} neighbors in group s [18, 23, 53, 15]. Our model only constrains the total number of edges within or between groups.

As a consequence, the Lovász ϑ function cannot refute k-colorability with high probability if

$$k > 2 + \frac{d}{2\sqrt{d-1}}\,, \tag{5}$$

and in particular if d is below the Kesten-Stigum threshold.

A strict inequality suffices in (5) by appropriately choosing ϵ in (4). Rearranging, no refutation of this kind can exist when

$$d < 2(k-2)\left((k-2) + \sqrt{(k-2)^2 - 1}\right) = (4 - o_k(1))d_{\mathrm{KS}}\,.$$

Our lower bound on $\vartheta(\overline{G_{n,d}})$ follows easily from Friedman's theorem [29] on the spectrum of $G_{n,d}$. For the upper bound, we first use orthogonal polynomials to derive explicit bounds on $\vartheta(\overline{G})$ for arbitrary regular graphs of a given girth – which may be of independent interest – and then employ a concentration argument for $G_{n,d}$.

We also relate the Lovász ϑ function to the existence of a good partition in the disassortative case of the block model, giving

▶ **Theorem 2.** *Fix $\tau < 1$ and say a partition is* good *if a fraction τ/k of its edges connect endpoints in the same group. Then sum-of-squares proofs of degree two cannot refute the existence of a good partition in $G_{n,d}$ if*

$$\frac{k-\tau}{1-\tau} > 2 + \frac{d}{2\sqrt{d-1}}\,.$$

Thus degree-two sum-of-squares cannot distinguish the regular stochastic block model from $G_{n,d}$ until d is roughly a factor of 4 above the Kesten-Stigum threshold.

Related Work

The distribution of $\vartheta(\overline{G})$ for the Erdős-Rényi graph $G = G(n,p)$ and the random d-regular graph $G = G_{n,d}$ were studied in [20]. In particular, that work showed that when d is sufficiently large, with high probability $\vartheta(\overline{G_{n,d}}) > c\sqrt{d}$ for a constant $c > 0$. Our results tighten this lower bound, making the constant c explicit, and provide a nearly-matching upper bound.

Our results on the power of degree-two sum-of-squares refutations for k-colorability contribute to a recent line of work on refutations of random CSPs, which we briefly survey. If we define the density of a CSP as the ratio of constraints to variables – which for coloring equals half the average degree of the graph – then the conjectured hard regime for k-coloring corresponds to a range of densities bounded below and above by constants (i.e., depending on k but not n). For CSPs such as k-SAT and k-XOR, there is again a satisfiability transition at constant density, but with high probability sum-of-squares refutations with constant degree do not exist unless the density is much higher, namely $\Omega(n^{k/2-1})$ [55], a result which was recently extended to general CSPs whose constraint predicate supports a $(k-1)$-wise uniform distribution [36]. Conversely, if a predicate does not support a t-wise uniform distribution, then [10] shows that there is an efficient sum-of-squares refutation when the density is $\tilde{O}(n^{t/2} - 1)$. For coloring, this gives refutations at roughly constant density; our contribution makes this a nearly-precise constant in the special case of degree-two sum-of-squares on random regular graphs.

The hidden clique problem also has a conjectured hard regime. It is well known that the random graph $G(n, 1/2)$ has no cliques larger than $O(\log n)$ [28] but it is conjectured to be computationally hard to distinguish $G(n, 1/2)$ from a graph with a planted clique of size $o(n^{1/2})$. A sequence of progressively stronger sum-of-squares lower bounds for this problem [27, 32, 45] have culminated in the theorem that with high probability the degree-d sum-of-squares proof system cannot refute the existence of a clique of size $n^{1/2-c(d/\log n)^{1/2}}$ in $G(n,1/2)$ for some constant $c>0$ [13].

In contrast to the aforementioned work on refuting random k-CSPs and planted cliques, our result pertains to a much more specific pair of problems, namely k-coloring and the stochastic block model, and only to degree-two sum-of-squares refutations; but it attains a sharp bound, within an additive constant, on the density at which these refutations become possible. We conjecture that sum-of-squares refutations of any constant degree do not exist below the Kesten-Stigum threshold, but it seems difficult to extend our current techniques to degree higher than two.

2 Colorings, Partitions, and the Lovász ϑ Function

2.1 Background on sum-of-squares

One type of refutation which has gained a great deal of interest recently is sum-of-squares proofs: see [14] for a review. Suppose we encode our variables and constraints as a system of m polynomial equations on n variables, $f_j(x_1,x_2,\ldots,x_n)=0$ for all $j=1,\ldots,m$.

One way to prove that no solution $\boldsymbol{x}\in\mathbb{R}^n$ exists – in algebraic terms, that this variety is empty – is to find a linear combination of the f_j which is greater than zero for all $\boldsymbol{x}$. Moreover, the *positivstellensatz* of Krivine [37] and Stengle [57] shows that a polynomial is nonnegative over $\mathbb{R}^n$ if and only if it can be written as a sum of squares of rational functions. Thus, clearing denominators, we need polynomials $g_1,\ldots,g_m$ and $h_1,\ldots,h_t$ and a constant $\epsilon>0$ (which we can always scale to 1 if we like) such that

$$\sum_{j=1}^{m} g_j(\boldsymbol{x})f_j(\boldsymbol{x}) = S+\epsilon \quad \text{where} \quad S=\sum_{\ell=1}^{t} h_\ell(\boldsymbol{x})^2\,. \tag{6}$$

This proof technique is complete as well as sound. That is, there is such a set of polynomials $\{g_j\}$ and $\{h_\ell\}$ if and only if no solution exists.

Even when the f_j are of low degree, the polynomials g_j and h_ℓ might be of high degree, making them difficult to find. However, we can ask when a refutation exists where both sides of (6) have degree δ or less. As we take $\delta=2,4,6,\ldots$ we obtain the *SOS hierarchy*. The case $\delta=2$ is typically equivalent to a familiar semidefinite relaxation of the problem. More generally, a degree-δ refutation exists if and only if a certain semidefinite program on $O(n^\delta)$ variables is feasible: thus we can find degree-δ refutations, or confirm that they do not exist, in time poly(n^δ) [56, 52, 54, 41]. To see why, note that if we write a polynomial $S(\boldsymbol{x})$ as a bilinear form on monomials $x^{(\alpha)}=\prod_i x^{\alpha_i}$ of degree $\delta/2$,

$$S(\boldsymbol{x})=\sum_{\alpha,\alpha'}\mathcal{S}(\alpha,\alpha')\,x^{(\alpha)}x^{(\alpha')}\,,$$

then $S(\boldsymbol{x})$ is a sum of squares of degree $\delta/2$ polynomials if and only if the matrix $\mathcal{S}$ is positive semidefinite, or equivalently if $\mathcal{S}$ is the sum of positive symmetric rank-one matrices. These are outer products of vectors with themselves, so there are vectors $w_1,\ldots,w_t$ such that $\mathcal{S}=\sum_{\ell=1}^t w_\ell\otimes w_\ell$ and $S=\sum_\ell h_\ell^2$ where $h_\ell(\boldsymbol{x})=\sum_\alpha w_\ell(\alpha)x^{(\alpha)}$. Finally, the constraint that $S=\sum_j g_jf_j-\epsilon$ for some $\{g_j\}$ and some $\epsilon>0$ corresponds to a set of linear inequalities on the entries of $\mathcal{S}$.

The dual object to a degree-δ refutation is a *pseudoexpectation.* This is a linear operator $\tilde{\mathbb{E}}$ on polynomials of degree at most δ with the properties that

$$\tilde{\mathbb{E}}[1] = 1, \tag{7}$$

$$\tilde{\mathbb{E}}[f_j] = 0 \text{ for all } j, \tag{8}$$

$$\tilde{\mathbb{E}}[p^2] \geq 0 \text{ for any polynomial } p \text{ of degree at most } \delta/2. \tag{9}$$

If we write $\tilde{\mathbb{E}}$ as a bilinear form on monomials $x^{(\alpha)}$, then (7) and (8) are linear constraints on its entries, and (9) states that this matrix is positive semidefinite. The resulting SDP is dual to the SDP for refutations, so each of these SDPs is feasible precisely when the other is not. Thus there is a degree-δ refutation if and only if no degree-δ pseudoexpectation exists, and vice versa.

We can think of a pseudoexpectation as a way for an adversary to fool the SOS proof system. The adversary claims there are are many solutions – even if in reality there are none – and offers to compute the expectation of any low-degree polynomial over the set of solutions. As long as (7) and (8) hold, this appears to be a distribution over valid solutions, and as long as (9) holds, the SOS prover cannot catch the adversary in an obvious lie like the claim that some quantity of degree $\delta/2$ has negative variance.

2.2 Colorings, partitions, and sum-of-squares

For a given graph G with adjacency matrix A, we can encode the problem of k-colorability as the following system of polynomial equations in kn variables $\boldsymbol{x} = \{x_{i,c}\}$, where $i \in [n]$ indexes vertices and $c \in [k]$ indexes colors:

$$\text{The } x_{i,c} \text{ are Boolean:} \qquad p^{\text{bool}}_{i,c} \triangleq x^2_{i,c} - x_{i,c} = 0 \qquad \forall i, c \tag{10}$$

$$\text{Each vertex has one color:} \qquad p^{\text{sing}}_{i} \triangleq -1 + \sum_c x_{i,c} = 0 \qquad \forall i \tag{11}$$

$$\text{The coloring is proper:} \qquad p^{\text{col}}_{ij} \triangleq \sum_c x_{i,c}\, x_{j,c} = 0 \qquad \forall (i,j) \in E \tag{12}$$

Then G is k-colorable if and only if (10)–(12) has a solution in $\mathbb{R}^{kn}$. We can encode the stochastic block model similarly: fix τ, and recall that a partition of G into k groups is *good* if a fraction τ/k of the edges have endpoints in the same group. If G has m edges, we can replace constraint (12) with

$$\text{Good partition:} \qquad p^{\text{cut}} \triangleq -\frac{\tau}{k} + \frac{1}{2m} \sum_{i,j} A_{ij} \sum_c x_{i,c}\, x_{j,c} = 0\,. \tag{13}$$

A degree-δ sum-of-squares refutation of (10)–(12) is an equation of the form

$$\sum_{i,c} b_{i,c} p^{\text{bool}}_{i,c} + \sum_i s_i p^{\text{sing}}_i + \sum_{(i,j)\in E} g_{ij} p^{\text{col}}_{ij} = S + \epsilon \tag{14}$$

where $b_{i,c}, s_i, g_{ij}$ are polynomials over $\boldsymbol{x}$, S is a sum of squares of polynomials, ϵ is a small positive constant which we will omit when clear, and the degree of each side is at most δ. Such an equation is a proof that no coloring exists. Replacing $\sum_{i,j} g_{ij} p^{\text{col}}_{ij}$ with $g_{\text{cut}} p^{\text{cut}}$ gives a refutation of the system formed by (10), (11), and (13), proving that no good partition exists. We focus on refutations of degree two, which as we will see are related to a classic relaxation of graph coloring.

2.3 The Lovász ϑ function

An *orthogonal representation* of a graph G with n vertices is an assignment of a unit vector $u_i \in \mathbb{R}^n$ to each vertex i such that $\langle u_i, u_j \rangle = 0$ for all $(i,j) \in E$. The Lovász function, denoted $\vartheta(\overline{G})$ by convention, is the smallest κ for which there is an orthogonal representation $\{u_i\}$ and an additional unit vector $\mathfrak{z} \in \mathbb{R}^n$ such that $\langle u_i, \mathfrak{z} \rangle = 1/\sqrt{\kappa}$: that is, such that all the u_i lie on a cone[2] of width $\cos^{-1}(1/\sqrt{\kappa})$.

The Gram matrix $P_{ij} = \langle u_i, u_j \rangle$ of an orthogonal representation is positive semidefinite with $P_{ii} = 1$ and $P_{ij} = 0$ for $(i,j) \in E$. Adding an auxiliary row and column for the inner products with $\mathfrak{z}$, we can define ϑ in terms of a semidefinite program,

$$\vartheta(\overline{G}) = \min_P \kappa > 0 \qquad \text{such that} \qquad \begin{aligned} \begin{pmatrix} 1 & \mathbf{1}/\sqrt{\kappa} \\ \mathbf{1}/\sqrt{\kappa} & P \end{pmatrix} &\succeq 0 \\ P_{ii} &= 1 \qquad \forall i \\ P_{ij} &= 0 \qquad \forall (i,j) \in E\,, \end{aligned} \tag{15}$$

where $\mathbf{1}$ is the n-dimensional vector whose entries are all 1s. The dual of this program can be written as

$$\vartheta(\overline{G}) = \max_D \langle D, \mathbb{J} \rangle \qquad \text{such that} \qquad \begin{aligned} D &\succeq 0 \\ \operatorname{tr} D &= 1 \\ D_{ij} &= 0 \qquad \forall (i,j) \notin E\,, \end{aligned} \tag{16}$$

where $\mathbb{J}$ is the matrix of all 1s and $\langle A, B \rangle = \operatorname{tr}(A^\dagger B) = \sum_{i,j} A_{ij} B_{ij}$ denotes the matrix inner product.

If G is k-colorable then $\vartheta(\overline{G}) \le k$, since we can use the first k basis vectors $e_1, \ldots, e_k$ as an orthogonal representation and take $\mathfrak{z} = (1/\sqrt{k}) \sum_{t=1}^k e_t$. Thus if $\vartheta(\overline{G}) > k$, the Lovász function gives a polynomial-time refutation of k-colorability. As stated above, degree-two sum-of-squares proofs typically correspond to well-known semidefinite relaxations, and the next theorem shows that this is indeed the case here.

▶ **Theorem 3.** *There is a degree-2 SOS refutation of k-colorability for a graph G if and only if $\vartheta(\overline{G}) > k$.*

We prove this in the Appendix, where we show that any orthogonal representation of G that lies on an appropriate cone lets us define a pseudoexpectation for the system (10)–(12). This will also allow us to modify the SDPs for refutations and pseudoexpectations, and work with simplified but equivalent versions.

2.4 Good partitions and a relaxed Lovász function

The reader may have noticed that while the coloring constraint (12) fixes the inner product $\sum_c x_{i,c} x_{j,c} = \langle x_i, x_j \rangle$ to zero for each edge $(i,j) \in E$, the "good partition" constraint (13) only fixes the sum of all these inner products. This suggests a slight relaxation of the Lovász ϑ function, where we weaken the SDP (15) by replacing the individual constraints on P_{ij} for

[2] To see that this definition of ϑ is equivalent to the more common one that $\langle u_i, \mathfrak{z} \rangle \le 1/\sqrt{\kappa}$ for every i, i.e., where the u_i can be in the interior of this cone, simply rotate each u_i in the subspace perpendicular to its neighbors until $\langle u_i, \mathfrak{z} \rangle$ is exactly $1/\sqrt{\kappa}$.

all $(i,j) \in E$ with a constraint on their sum. In other words, we allow a vector coloring where neighboring vectors are orthogonal on average. We denote the resulting function $\hat{\vartheta}$:

$$\hat{\vartheta}(\overline{G}) = \min_P \kappa > 0 \qquad \text{such that} \qquad \begin{pmatrix} 1 & \mathbf{1}/\sqrt{\kappa} \\ \mathbf{1}/\sqrt{\kappa} & P \end{pmatrix} \succeq 0 \qquad (17)$$

$$P_{ii} = 1 \qquad \forall i$$

$$\langle P, A \rangle = 0 \,,$$

The dual SDP tightens (16) by requiring that the matrix D take the same value on every edge. Thus D is a multiple of A plus a diagonal matrix,

$$\hat{\vartheta}(\overline{G}) = \max_{\eta, \boldsymbol{b}} \langle D, \mathbb{J} \rangle \qquad \text{such that} \qquad D \triangleq \eta A + \operatorname{diag} \boldsymbol{b} \succeq 0 \qquad (18)$$

$$\operatorname{tr} D = \langle \boldsymbol{b}, \mathbf{1} \rangle = 1$$

Since $\hat{\vartheta}$ is a relaxation of ϑ, we always have $\hat{\vartheta}(\overline{G}) \le \vartheta(\overline{G})$.

This modified Lovász function $\hat{\vartheta}$ is equivalent to degree-two SOS for good partitions in the dissassortative case of the block model, in the following sense.

▶ **Theorem 4.** *If $\tau < 1$, there exists a degree-two SOS refutation of a partition of G where a fraction τ/k of the edges are within groups if and only if*

$$\hat{\vartheta}(\overline{G}) > \frac{k-\tau}{1-\tau} \,. \qquad (19)$$

Once again we leave the proof to the Appendix. Note that the SDP (17) for $\hat{\vartheta}$ contains no information about k or τ: this relaxed orthogonal representation has the uncanny capacity to fool degree-two SOS about an entire family of related cuts of different sizes and qualities.

2.5 Upper and lower bounds

With these theorems in hand, we can set about producing degree-two sum-of-squares refutations and pseudoexpectations for our problems; throughout this section we will refer to these simply as 'refutations' and 'pseudoexpectations'. In fact, the same construction will give us asymptotically optimal refutations and pseudoexpectations for both the coloring and partition problems.

To warm-up, we have the following simple construction of a refutation, which we will phrase in terms of the Lovász theta function and its relaxed version.

▶ **Lemma 5.** *Let G be a d-regular graph, and let $\lambda_{\min}$ be the smallest eigenvalue of its adjacency matrix A. Then*

$$\vartheta(\overline{G}) \ge \hat{\vartheta}(\overline{G}) \ge 1 + d/|\lambda_{\min}| \,. \qquad (20)$$

Proof. Denote by $\mathbb{1}$ the identity matrix. We construct a feasible solution D to the dual SDP (18) by taking

$$D \triangleq \frac{1}{n}\left(\mathbb{1} + \frac{1}{|\lambda_{\min}|} A\right) ,$$

and use the fact that $\langle A, \mathbb{J} \rangle = dn$. ◀

By invoking Friedman's theorem [29] that (as $n \to \infty$) the smallest eigenvalue of a random d-regular graph is with high probability larger than $-2(1+\epsilon)\sqrt{d-1}$ for any $\epsilon > 0$, we obtain:

▶ **Corollary 6.** *When $G = G_{n,d}$, for any $\epsilon > 0$, with high probability*

$$\vartheta(\overline{G}) \geq \hat{\vartheta}(\overline{G}) > 1 + \frac{d}{2\sqrt{d-1}} - \epsilon\,. \tag{21}$$

Putting this together with Theorems 3 and 4 gives

▶ **Corollary 7.** *If $G = G_{n,d}$ and $\tau < 1$, with high probability there exists a refutation of a partition with a fraction τ/k of within-group edges when*

$$\frac{k-\tau}{1-\tau} < 1 + \frac{d}{2\sqrt{d-1}}\,. \tag{22}$$

Setting $\tau = 0$, a refutation of k-colorability exists with high probability when

$$k < 1 + \frac{d}{2\sqrt{d-1}}\,.$$

Note that for large k, the minimum value of d satisfying (22) is a factor of four above the Kesten-Stigum threshold in both the coloring and partition problems.

Our construction for this lower bound on ϑ is quite simple, but remarkably we find that for both the coloring and partition problems, it is asymptotically optimal in d and k. In particular,

▶ **Theorem 8.** *For any d-regular graph G with girth at least γ, we have*

$$\hat{\vartheta}(\overline{G}) \leq \vartheta(\overline{G}) < 1 + \frac{d}{2(1-\epsilon_\gamma)\sqrt{d-1}}\,. \tag{23}$$

where ϵ_γ is a sequence of constants which decrease to zero as $\gamma \to \infty$.

Since for any constant γ a random regular graph has girth γ with positive probability [59, Theorem 2.12], we rely on the following result showing that $\vartheta(\overline{G_{n,d}})$ is concentrated in an interval of width one. The proof is essentially the same as that of [7] for the chromatic number, and is given in the Appendix.

▶ **Lemma 9.** *Let $\theta \geq 3$. If $\vartheta(\overline{G_{n,d}}) \leq \theta$ with positive probability, then $\vartheta(\overline{G_{n,d}}) \leq \theta + 1$ with high probability.*

▶ **Corollary 10.** *If $G = G_{n,d}$, with high probability there does not exist a refutation of a partition with a fraction τ/k of within-group edges when*

$$\frac{k-\tau}{1-\tau} > 2 + \frac{d}{2\sqrt{d-1}}\,. \tag{24}$$

Setting $\tau = 0$, with high probability no refutation of k-colorability exists when

$$k > 2 + \frac{d}{2\sqrt{d-1}}\,.$$

Thus for both problems, no degree-two sum-of-squares refutation exists until d is roughly a factor of 4 above the Kesten-Stigum threshold.

3 Constructing a Pseudoexpectation with Orthogonal Polynomials

We now prove Theorem 8 by constructing a feasible solution to the primal SDP (15): that is, unit vectors $\{u_i\}$ such that $\langle u_i, u_j \rangle = 0$ for every edge (i, j), and a unit vector $\mathfrak{z}$ so that $\langle u_i, \mathfrak{z} \rangle = 1/\sqrt{\kappa}$ for all i. Recall that such a collection exists if and only if $\vartheta(\overline{G}) \leq \kappa$.

It is convenient to instead define a set of unit vectors $\{v_i\}$ such that $\langle v_i, v_j \rangle = -1/(\kappa - 1)$ for every edge (i, j). We claim that such a set exists if and only if $\vartheta(\overline{G}) \leq \kappa$. In one direction, given $\{u_i\}$ and $\mathfrak{z}$ with the above properties, if we define

$$v_i = \sqrt{\frac{\kappa}{\kappa - 1}}\, u_i - \frac{1}{\sqrt{\kappa - 1}}\, \mathfrak{z}$$

then the v_i are unit vectors with $\langle v_i, v_j \rangle = -1/(\kappa - 1)$ for $(i, j) \in E$. For instance, if the u_i are k orthogonal basis vectors, then the v_i point to the corners of a k-simplex. In the other direction, given $\{v_i\}$ we can take $\mathfrak{z}$ to be a unit vector perpendicular to all the v_i, and define

$$u_i = \sqrt{\frac{\kappa - 1}{\kappa}}\, v_i + \frac{1}{\sqrt{\kappa}}\, \mathfrak{z}\,.$$

Then $\langle u_i, u_j \rangle = 0$ for $(i, j) \in E$, and $\langle u_i, \mathfrak{z} \rangle = 1/\sqrt{\kappa}$ for all i. This means that we can characterize the Lovász ϑ function with a slightly different SDP, which uses the Gram matrix of the $\{v_i\}$:

$$\begin{aligned} \vartheta(\overline{G}) = \min_P \kappa > 1 \qquad \text{such that} \qquad & P \succeq 0 & & \qquad (25)\\ & P_{ii} = 1 & \forall i & \\ & P_{ij} = -1/(\kappa - 1) & \forall (i, j) \in E & \end{aligned}$$

Alternatively, the matrix P above is a scaled Schur complement of the block matrix in (15).

We will show that for any d-regular graph G with girth at least γ, this SDP has a feasible solution with

$$\kappa = 1 + \frac{d}{2(1 - \epsilon_\gamma)\sqrt{d - 1}}\,,$$

where ϵ_γ depends only on γ and tends to zero as $\gamma \to \infty$. Therefore, there is a pseudoexpectation that prevents degree-two SOS from refuting k-colorability for any $k \geq \kappa$. We will construct this pseudoexpectation by taking a linear combination of the "non-backtracking powers" of G's adjacency matrix A.

Denote by $A^{(t)}$ the matrix whose i, j entry is the number of non-backtracking walks of length t from i to j; that is, walks which may freely wander the graph so long as they do not make adjacent pairs of steps $a \to b \to a$ for any vertices a, b. There is a simple two-term recursion for these matrices: to count non-backtracking walks of length $t + 1$, we first extend each walk of length t by one edge, and then subtract off those that backtracked on the last step. This gives

$$\begin{aligned} A^{(0)} &= \mathbb{1} \\ A^{(1)} &= A \\ A^{(2)} &= A^2 - d\mathbb{1} \\ A^{(t)} &= A \cdot A^{(t-1)} - (d - 1)A^{(t-2)} \quad t \geq 3\,. \end{aligned} \qquad (26)$$

Borrowing notation from [11], we can write $A^{(t)}$ in closed form as

$$A^{(t)} = \sqrt{d(d-1)^{t-1}}\, q_t\!\left(\frac{A}{2\sqrt{d-1}}\right) \qquad t \ge 1 \tag{27}$$

where $q_t(z)$ is a polynomial of degree t. Specifically,

$$q_0(z) = 1$$
$$q_1(z) = 2\sqrt{\frac{d-1}{d}}\,z$$

and for $t > 1$ the q_t satisfy the Chebyshev recurrence

$$q_{t+1}(z) = 2zq_t(z) - q_{t-1}(z)\,.$$

We can write q_t explicitly as

$$q_t(z) = \sqrt{\frac{d-1}{d}}\,U_t(z) - \frac{1}{\sqrt{d(d-1)}}\,U_{t-2}(z) \qquad t \ge 1 \tag{28}$$

and U_t is the tth Chebyshev polynomial of the second kind (note that $U_{-1}(z) = 0$).

Let $\mu(z)$ denote the Kesten-McKay measure μ on the interval $[-1, +1]$, which after scaling by $2\sqrt{d-1}$ describes the typical spectral density of a random regular graph [44]:

$$\mu(z) = \frac{2}{\pi}\left(\frac{d(d-1)}{d^2 - 4(d-1)z^2}\right)\sqrt{1-z^2}\,. \tag{29}$$

Then the polynomials q_t are orthonormal with respect to this measure. That is, if we define the inner product

$$\langle f, g\rangle = \int f(z)\,g(z)\,\mathrm{d}\mu = \int_{-1}^{1} f(z)\,g(z)\,\mu(z)\,\mathrm{d}z\,,$$

then

$$\langle q_\ell(z), q_m(z)\rangle = \begin{cases} 1 & \ell = m \\ 0 & \ell \ne m\,. \end{cases} \tag{30}$$

If the girth of the graph is at least γ, there is no way for a non-backtracking walk of length $\gamma - 2$ or less to return to its starting point or to a neighbor of its starting point, so $\left\langle \mathbb{1}, A^{(t)}\right\rangle = \left\langle A, A^{(t)}\right\rangle = 0$ for $1 < t \le \gamma - 2$. We can thus satisfy the diagonal and edge constraints of (25) by considering solutions of the form

$$\begin{aligned} P &= \mathbb{1} - \frac{1}{\kappa-1}A + \sum_{t=2}^{\gamma-2} a_t A^{(t)} \\ &= \mathbb{1} - \frac{\sqrt{d}}{\kappa-1}\,q_1\!\left(\frac{A}{2\sqrt{d-1}}\right) + \sum_{t=2}^{\gamma-2} a_t\sqrt{d(d-1)^{t-1}}\,q_t\!\left(\frac{A}{2\sqrt{d-1}}\right) \\ &\triangleq f\!\left(\frac{A}{2\sqrt{d-1}}\right)\,, \end{aligned} \tag{31}$$

since the first two terms ensure that P has 1s on its diagonal and $-1/(\kappa-1)$ on the edges. If we write

$$f(z) = \sum_{t=0}^{\gamma-2} c_t q_t(z) \quad \text{where} \quad c_0 = 1 \quad \text{and} \quad c_1 = -\frac{\sqrt{d}}{\kappa-1}\,, \tag{32}$$

our job is to optimize the coefficients c_t for $1 < t \leq \gamma - 2$ so as to minimize c_1, and hence κ, while ensuring that $P \succeq 0$.

The eigenvalues of the matrix $f(A/(2\sqrt{d-1}))$ are of the form $f(\lambda/(2\sqrt{d-1}))$ where λ ranges over all of A's eigenvalues. Therefore, $P \succeq 0$ if and only if $f(\lambda/(2\sqrt{d-1}))$ for all eigenvalues λ of A. Friedman's celebrated theorem [29] shows that, with high probability, the eigenvalues of A are contained in the set

$$S = \Big(-(1+\epsilon)2\sqrt{d-1},\, (1+\epsilon)2\sqrt{d-1}\Big) \cup \{d\}$$

for any $\epsilon > 0$. Thus we require that

$$f(z) \geq 0 \quad \text{for all} \quad z \in \big(-(1+\epsilon), 1+\epsilon\big) \cup \left\{ \frac{d}{2\sqrt{d-1}} \right\} . \tag{33}$$

We will relax this condition slightly by demanding just that f is nonnegative on $[-1, +1]$, although as we will see the resulting optimum is achieved by a function which is nonnegative on all of $\mathbb{R}$. First we use orthonormality (30) to write the coefficients c_t as inner products,

$$c_t = \langle q_t, f \rangle .$$

Then we optimize the pseudoexpectation as follows,

$$\begin{aligned} \min \quad & \langle q_1, f \rangle \\ \text{such that} \quad & \langle q_0, f \rangle = 1 \\ & f(z) \geq 0 \qquad \forall z \in [-1, +1] . \end{aligned} \tag{34}$$

When the degree $\gamma - 2$ of f is even, we can solve this optimization problem explicitly. Set $m = \gamma/2$, and let $r_1 > \cdots > r_m$ be the roots of q_m in decreasing order; it follows from standard arguments about orthogonal polynomials that these are all in the support of μ, i.e., in the interval $[-1, +1]$. Consider the following polynomial of degree $2(m-1) = \gamma - 2$,

$$s(z) = \frac{1}{\zeta} \prod_{j=1}^{m-1} (z - r_j)^2 , \tag{35}$$

where

$$\zeta = \left\langle q_0, \prod_{j=1}^{m-1} (z - r_j)^2 \right\rangle$$

is a normalizing factor to ensure that $\langle q_0, s \rangle = 1$. We claim that $s(z)$ is the optimum of (34). To prove this, we begin with a general lemma on orthogonal polynomials and quadrature. The proof is standard (e.g. [58]) but we include it in the Appendix for completeness.

▶ **Lemma 11.** *Let $\{p_t\}$ be a sequence of polynomials of degree t which are orthogonal with respect to a measure ρ supported on a compact interval I. Then the roots $r_1, \ldots, r_t$ of p_t form a quadrature rule which is exact for any polynomial u of degree less than $2t$, in that*

$$\int_I u(z)\, \mathrm{d}\rho = \sum_{i=1}^{t} \omega_i u(r_i)$$

for some positive weights $\{\omega_1, \ldots, \omega_t\}$ independent of u.

Now let $g(z) = z - r_m$. In view of Lemma 11, for any polynomial $f(z)$ of degree at most $\gamma - 2$, the inner product $\langle g, f \rangle$ can be expressed using the roots $r_1, \ldots, r_m$ of q_m as a quadrature,

$$\langle g, f \rangle = \int (z - r_m) f(z) \, \mathrm{d}\mu = \sum_{j=1}^{m} \omega_j (r_j - r_m) f(r_j) = \sum_{j=1}^{m-1} \omega_j (r_j - r_m) f(r_j) \, .$$

Note that $\omega_j(r_j - r_m) > 0$ for every $1 \le j \le m-1$, since r_m is the left-most root. If impose the constraints that $f(r_j) \ge 0$ for all $j = 1, \ldots, m-1$, then $\langle g, f \rangle \ge 0$. If we also impose the constraint $\langle f, q_0 \rangle = 1$, then

$$\begin{aligned}\langle q_1, f \rangle &= \left\langle 2\sqrt{\frac{d-1}{d}}\, z, f \right\rangle \\ &= 2\sqrt{\frac{d-1}{d}}\, \langle g, f \rangle + 2\sqrt{\frac{d-1}{d}}\, r_m \langle q_0, f \rangle \\ &\ge 2\sqrt{\frac{d-1}{d}}\, r_m \, , \end{aligned} \tag{36}$$

with equality if and only if $f(r_j) = 0$ for all $j = 1, \ldots, m-1$. Since $s(z)$ obeys this equality condition, we have

$$\langle q_1, s \rangle = 2\sqrt{\frac{d-1}{d}}\, r_m \, ,$$

and this is the minimum possible value of $c_1 = \langle q_1, s \rangle$ subject to the constraints that $\langle q_0, f \rangle = 1$ and $f(r_j) \ge 0$ for $j = 1, \ldots, m-1$. Moreover, $s(z) \ge 0$ on all of $\mathbb{R}$, so $s(z)$ in fact obeys the stronger constraint (33).

Referring back to (32) gives

$$c_1 = -\frac{\sqrt{d}}{\kappa - 1} = 2\sqrt{\frac{d-1}{d}}\, r_m \, ,$$

and so

$$\vartheta \ge \kappa = 1 + \frac{d}{2(-r_m)\sqrt{d-1}} \, .$$

Finally, we obtain (23) by defining $\epsilon_\gamma = r_m + 1$. Since $r_m \to -1$ as m tends to infinity[3], we have $\epsilon_\gamma \to 0$ as $\gamma \to \infty$, completing the proof.

We end with a brief note on the above construction. Recall that our project for the last several pages has been to set the coefficients of non-backtracking paths of length t in a feasible solution P to the SDP (25),

$$P = \sum_{t=0}^{\gamma} a_t A^{(t)} .$$

As discussed in the Appendix, this matrix can be translated into a degree-two pseudoexpectation $\tilde{\mathbb{E}}$ for the coloring problem: a linear operator that claims to give the joint distribution

[3] The fact that $r_m \to -1$ as $m \to \infty$ can be deduced, for example, by using the definition of q_m in (28) to observe that $q_m(-1)$ and $q_m(-\cos(\frac{\pi}{m-1}))$ have opposite signs, and then applying the intermediate value theorem.

of colors at at each pair of vertices i and j. The reader will find there that P_{ij} is related to the 'pseudocorrelation' between vertices i and j, by

$$\frac{k}{k-1}\left(P_{ij} - 1/k\right) = \widetilde{\Pr}[i \text{ and } j \text{ are the same color}] .$$

Our expansion of P in terms of non-backtracking paths means that, for most pairs i, j, this pseudoexpectation depends only on the shortest path distance $d(i, j)$. Specifically, whenever $d(i, j) = t \leq \gamma - 2$ and the shortest path is unique, we have $P_{ij} = a_t$, and if $d(i, j) > \gamma - 2$ then $P_{ij} = 0$. One might think that in the limit of large γ, the optimal pseudoexpectation would make the natural choice that $a_t = (1-k)^{-t}$: in that, case, the pseudocorrelation would decay just as if these shortest paths were colored uniformly at random, ignoring correlations with the remainder of the graph. However, a quick calculation shows that this choice is in fact not optimal. In fact, the optimal coefficients we derive above cause the pseudocorrelation to decay more quickly with distance than this naïve guess.

Acknowledgements. We are grateful to Charles Bordenave, Emmanuel Abbe, Amin Coja-Oghlan, Yash Deshpande, Marc Lelarge, and Alex Russell for helpful conversations. Part of this work was done while C.M. was visiting École Normale Supérieure. Part of this work was done while R.K. was a researcher at Microsoft Research New England.

References

1 E. Abbe, A. S. Bandeira, and G. Hall. Exact recovery in the stochastic block model. *IEEE Transactions on Information Theory*, 62(1):471–487, 2016.

2 Emmanuel Abbe. Community detection and stochastic block models: recent developments. *J. Machine Learning Research*, 2017. to appear.

3 Emmanuel Abbe and Colin Sandon. Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery. In *Proc. 56th Annual Symposium on Foundations of Computer Science, FOCS*, pages 670–688, 2015.

4 Emmanuel Abbe and Colin Sandon. Detection in the stochastic block model with multiple clusters: proof of the achievability conjectures, acyclic BP, and the information-computation gap. *ArXiv preprints*, 1512.09080, 2015. URL: http://arxiv.org/abs/1512.09080.

5 Emmanuel Abbe and Colin Sandon. Achieving the KS threshold in the general stochastic block model with linearized acyclic belief propagation. In *Proc. Neural Information Processing Systems (NIPS)*, pages 1334–1342, 2016. URL: http://papers.nips.cc/paper/6365-achieving-the-ks-threshold-in-the-general-stochastic-block-model-with-linearized-acyclic-belief-propagation.

6 Emmanuel Abbe and Colin Sandon. Crossing the KS threshold in the stochastic block model with information theory. In *IEEE Intl. Symp. on Information Theory, ISIT*, pages 840–844, 2016. doi:10.1109/ISIT.2016.7541417.

7 Dimitris Achlioptas and Cristopher Moore. The chromatic number of random regular graphs. In *Proc. 8th International Workshop on Randomization and Computation (RANDOM)*, pages 219–228, 2004.

8 Dimitris Achlioptas and Assaf Naor. The two possible values of the chromatic number of a random graph. *Ann. Math.*, 162:1335–1351, 2005.

9 Naman Agarwal, Afonso S. Bandeira, Konstantinos Koiliaris, and Alexandra Kolla. Multisection in the stochastic block model using semidefinite programming. *ArXiv preprints*, 1507.02323, 2015. URL: http://arxiv.org/abs/1507.02323.

10 Sarah R. Allen, Ryan O'Donnell, and David Witmer. How to refute a random CSP. In *IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 689–708, 2015.

11 Noga Alon, Itai Benjamini, Eyal Lubetzky, and Sasha Sodin. Non-backtracking random walks mix faster. *Communications in Contemporary Mathematics*, 9(04):585–603, 2007.

12 Jess Banks, Cristopher Moore, Joe Neeman, and Praneeth Netrapalli. Information-theoretic thresholds for community detection in sparse networks. In *Proc. 29th Conference on Learning Theory, COLT*, pages 383–416, 2016. URL: http://jmlr.org/proceedings/papers/v49/banks16.html.

13 Boaz Barak, Samuel B. Hopkins, Jonathan Kelner, Pravesh Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. In *Proc. 57th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 428–437, 2016.

14 Boaz Barak and David Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:59, 2014. URL: https://eccc.weizmann.ac.il/report/2014/059/.

15 P. Barucca. Spectral partitioning in random regular blockmodels. *ArXiv e-prints*, 1610.02668, 2016. URL: https://arxiv.org/abs/1610.02668.

16 Peter J. Bickel and Aiyou Chen. A nonparametric view of network models and Newman–Girvan and other modularities. *Proc. Natl. Acad. Sci. USA*, 106:21068–21073, 2009.

17 Charles Bordenave, Marc Lelarge, and Laurent Massoulié. Non-backtracking spectrum of random graphs: Community detection and non-regular Ramanujan graphs. In *Proc. 56th Annual Symposium on Foundations of Computer Science, FOCS*, pages 1347–1357, 2015.

18 Gerandy Brito, Ioana Dumitriu, Shirshendu Ganguly, Christopher Hoffman, and Linh V. Tran. Recovery and rigidity in a regular stochastic block model. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1589–1601, 2016.

19 A. Coja-Oghlan, C. Efthymiou, N. Jaafari, M. Kang, and T. Kapetanopoulos. Charting the replica symmetric phase. *ArXiv preprints*, 1704.01043, 2017. URL: https://arxiv.org/abs/1704.01043.

20 Amin Coja-Oghlan. The Lovász number of random graphs. In *7th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 228–239, 2003.

21 Amin Coja-Oghlan, Charilaos Efthymiou, and Samuel Hetterich. On the chromatic number of random regular graphs. *J. Comb. Theory, Ser. B*, 116:367–439, 2016.

22 Amin Coja-Oghlan, Florent Krzakala, Will Perkins, and Lenka Zdeborová. Information-theoretic thresholds from the cavity method. *Proc. 49th Annual ACM on Symposium on Theory of Computing, STOC*, 2017. URL: http://arxiv.org/abs/1611.00814.

23 Amin Coja-Oghlan, Elchanan Mossel, and Dan Vilenchik. A spectral approach to analysing belief propagation for 3-colouring. *Combinatorics, Probability and Computing*, 18(6):881–912, 2009.

24 Amin Coja-Oghlan and Dan Vilenchik. Chasing the K-colorability threshold. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 380–389, 2013.

25 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, 2011. doi:10.1103/PhysRevE.84.066106.

26 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. doi:10.1103/PhysRevLett.107.065701.

27 Yash Deshpande and Andrea Montanari. Improved sum-of-squares lower bounds for hidden clique and hidden submatrix problems. In *Proc. 28th Conference on Learning Theory, COLT*, pages 523–562, 2015.

28 Paul Erdős. Some remarks on the theory of graphs. *Bulletin of the American Mathematical Society*, 53(4):292–294, 1947.

29 Joel Friedman. *A Proof of Alon's Second Eigenvalue Conjecture and Related Problems.* American Mathematical Society, 2008.

30 B. Hajek, Y. Wu, and J. Xu. Achieving exact cluster recovery threshold via semidefinite programming. *IEEE Transactions on Information Theory*, 62(5):2788–2797, 2016.

31 B. Hajek, Y. Wu, and J. Xu. Achieving exact cluster recovery threshold via semidefinite programming: Extensions. *IEEE Trans. on Information Theory*, 62(10):5918–5937, 2016.

32 Samuel B. Hopkins, Pravesh Kothari, Aaron Henry Potechin, Prasad Raghavendra, and Tselil Schramm. On the integrality gap of degree-4 sum of squares for planted clique. In *Proc. of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1079–1095, 2016.

33 Graeme Kemkes, Xavier Pérez-Giménez, and Nicholas Wormald. On the chromatic number of random d-regular graphs. *Advances in Mathematics*, 223(1):300–328, 2010.

34 H. Kesten and B. P. Stigum. Additional limit theorems for indecomposable multidimensional Galton–Watson processes. *Ann. Math. Stat.*, 37:1463–1481, 1966.

35 H. Kesten and B. P. Stigum. Limit theorems for decomposable multi-dimensional Galton–Watson processes. *J. Math. Anal. Appl.*, 17:309, 1966.

36 Pravesh K. Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. In *Proc. 49th Annual ACM Symposium on Theory of Computing, STOC*, 2017.

37 J. L. Krivine. Anneaux préordonnés. *Journal d'Analyse Mathématique*, 12(1):307–326, 1964.

38 F. Krzakala, C. Moore, E. Mossel, J. Neeman, A. Sly, L. Zdeborová, and P. Zhang. Spectral redemption in clustering sparse networks. *Proc. Natl. Acad. Sci. USA*, 110(52):20935–20940, 2013. `doi:10.1073/pnas.1312486110`.

39 Florent Krzakala, Andrea Montanari, Federico Ricci-Tersenghi, Guilhem Semerjian, and Lenka Zdeborová. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proc. Natl. Acad. Sci. USA*, 104(25):10318–10323, 2007. `doi:10.1073/pnas.0703685104`.

40 Florent Krzakala and Lenka Zdeborová. Hiding quiet solutions in random constraint satisfaction problems. *Phys. Rev. Lett.*, 102:238701, 2009.

41 Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

42 Tomasz Łuczak. A note on the sharp concentration of the chromatic number of random graphs. *Combinatorica*, 11(3):295–297, 1991.

43 Laurent Massoulié. Community detection thresholds and the weak Ramanujan property. In *Proc. 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 694–703, 2014.

44 Brendan D. McKay. The expected eigenvalue distribution of a random labelled regular graph. *Linear Algebra and its Applications*, 40:203–216, 1981.

45 Raghu Meka, Aaron Potechin, and Avi Wigderson. Sum-of-squares lower bounds for planted clique. In *Proc. 47th Annual ACM on Symposium on Theory of Computing (STOC)*, pages 87–96, 2015.

46 M. Molloy and B. A. Reed. The chromatic number of sparse random graphs, 1992. Masters thesis, University of Waterloo.

47 Cristopher Moore. The computer science and physics of community detection: Landscapes, phase transitions, and hardness. *Bulletin of the EATCS*, 121:25–61, 2017.

48 Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *ArXiv preprints*, 1311.4115, 2013. URL: `http://arxiv.org/abs/1311.4115`.

49 Elchanan Mossel, Joe Neeman, and Allan Sly. Belief propagation, robust reconstruction and optimal recovery of block models. In *Proc. 27th Conference on Learning Theory, COLT*, pages 356–370, 2014. URL: `http://jmlr.org/proceedings/papers/v35/mossel14.html`.

50 Elchanan Mossel, Joe Neeman, and Allan Sly. Consistency thresholds for the planted bisection model. In *Proc. Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC*, pages 69–75, 2015. `doi:10.1145/2746539.2746603`.

51 Elchanan Mossel, Joe Neeman, and Allan Sly. Reconstruction and estimation in the planted partition model. *Probability Theory and Related Fields*, 162(3-4):431–461, 2015.

52 Yurii Nesterov. Squared functional systems and optimization problems. *High Performance Optimization*, 33:405–440, 2000.

53 M. E. J. Newman and Travis Martin. Equitable random graphs. *Phys. Rev. E*, 90:052824, 2014.

54 Pablo A. Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization.* PhD thesis, California Institute of Technology, 2000.

55 Grant Schoenebeck. Linear level Lasserre lower bounds for certain k-CSPs. In *Proc. 49th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 593–602, 2008.

56 N. Z. Shor. An approach to obtaining global extremums in polynomial mathematical programming problems. *Cybernetics*, 23(5):695–700, 1987.

57 Gilbert Stengle. A nullstellensatz and a positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207(2):87–97, 1974.

58 Gabor Szegő. *Orthogonal Polynomials.* American Mathematical Society, 1939.

59 N. Wormald. Models of random regular graphs. In *Surveys in Combinatorics*, pages 239–298. Cambridge University Press, 1999.

A Proof of Theorems 3 and 4

We prove Theorems 3 and 4 by directly simplifying the SDP that defines feasible degree-two pseudoexpectations. The first step is a broad result on the structure of these objects that applies to any set of constraints which includes the boolean (10) and single-color (11) constraints and is suitably symmetric; we then specialize to the coloring and partition problems.

Recall that a degree-two pseudoexpectation for a system of polynomials $f_j(\boldsymbol{x}) = 0$ is a linear operator $\tilde{\mathbb{E}} : \mathbb{R}[\boldsymbol{x}]_{\leq 2} \to \mathbb{R}$ which satisfies

- $\tilde{\mathbb{E}}[1] = 1$,
- $\tilde{\mathbb{E}}[f_j q] = 0$ for any polynomials f_j and q such that $\deg f_j q \leq 2$,
- $\tilde{\mathbb{E}}[p^2] \geq 0$ for any polynomial p with $\deg p^2 \leq 2$.

We can identify such objects with PSD $(nk+1) \times (nk+1)$ matrices of the form

$$\tilde{\mathbb{E}} = \begin{pmatrix} 1 & \boldsymbol{\ell}^\dagger \\ \boldsymbol{\ell} & \mathcal{E} \end{pmatrix} \tag{37}$$

where $\ell_{i,c} = \tilde{\mathbb{E}}[x_{i,c}]$ and $\mathcal{E}_{(i,c),(j,c')} = \tilde{\mathbb{E}}[x_{i,c}\, x_{j,c'}]$. It is useful to think of $\mathcal{E}$ as a block matrix, with a $k \times k$ block $\mathcal{E}_{ij}$ corresponding to each pair of vertices i, j. Consistency with the boolean and single-color constraints (10), (11) then controls the diagonal elements and row and colum sums of each of these blocks,

$$\mathcal{E}_{(i,c),(i,c)} = \tilde{\mathbb{E}}[x_{i,c}^2] = \tilde{\mathbb{E}}[x_{i,c}] = \ell_{i,c} \qquad \forall i \tag{38}$$

$$\sum_{c'} \mathcal{E}_{(i,c),(j,c')} = \sum_{c'} \tilde{\mathbb{E}}[x_{i,c}\, x_{j,c'}] = \tilde{\mathbb{E}}[x_{j,c}] = \ell_{i,c} \qquad \forall i,j \tag{39}$$

Moreover, each of our constraints is fixed under permutations of the colors, and $\tilde{\mathbb{E}}$ inherits this symmetry. That is the matrix carries with it a natural S_k action that simultaneously permutes $\tilde{\mathbb{E}}[x_{i,c}] \to \tilde{\mathbb{E}}[x_{i,\sigma(c)}]$ and $\tilde{\mathbb{E}}[x_{i,c}\, x_{j,c'}] \to \tilde{\mathbb{E}}[x_{i,\sigma(c)}\, x_{i,\sigma(c')}]$. This action preserves the spectrum of $\tilde{\mathbb{E}}$ as a matrix, as well as every hard constraint. By convexity, we may assume that $\tilde{\mathbb{E}}$ is stabilized under it, by beginning with an arbitrary pseudoexpectation and averaging over its orbit.

This assumption substantially constrains and simplifies $\tilde{\mathbb{E}}$. In particular we are free to (i) assume that $\ell_{i,c} = \tilde{\mathbb{E}}[x_{i,c}] = 1/k$ and (ii) assume that each $k \times k$ block in $\mathcal{E}$ has only two distinct values: one on the diagonal and the other off the diagonal. In other words, the pseudoexpectation claims that the marginal distribution of each vertex is uniform, and that joint marginal of any two vertices depends only on the probability that they have the same or different colors.

As a result, for each i, j we can assume that $\mathcal{E}_{ij}$ is a linear combination of the identity matrix $\mathbb{1}_k$ and the matrix $\mathbb{J}_k$ of all 1s, and that the row and column sums of $\mathcal{E}_{ij}$ are all $1/k$. In that case for each i, j we can write

$$\mathcal{E}_{ij} = \frac{1}{k-1}\left(P_{ij} - \frac{1}{k}\right)\left(\mathbb{1}_k - \frac{\mathbb{J}_k}{k}\right) + \frac{\mathbb{J}_k}{k^2} \tag{40}$$

for some P_{ij}, or equivalently that

$$\mathcal{E} = \frac{1}{k-1}(P - \mathbb{J}_n/k) \otimes \left(\mathbb{1}_k - \frac{\mathbb{J}_k}{k}\right) + \frac{\mathbb{J}_{nk}}{k^2} \tag{41}$$

for some $n \times n$ matrix P. Note that

$$\operatorname{tr} \mathcal{E}_{ij} = P_{ij}\,,$$

so (38) requires that $P_{ii} = 1$ for all i.

Since the pseudoexpectation (37) consists of $\mathcal{E}$ with an additional row and column, we consider the following lemma. We leave its proof as an exercise for the reader.

▶ **Lemma 12.** *For any matrix X, vector $\boldsymbol{v}$ and scalar $b > 0$,*

$$\begin{pmatrix} b & \boldsymbol{v}^\dagger \\ \boldsymbol{v} & X \end{pmatrix} \succeq 0$$

if and only if $X - (1/b)\boldsymbol{v} \otimes \boldsymbol{v} \succeq 0 \quad .$

Since $\boldsymbol{\ell}$ is the nk-dimensional vector whose entries are all $1/k$, we have $\boldsymbol{\ell} \otimes \boldsymbol{\ell} = \mathbb{J}_{nk}/k^2$. Thus (41) and Lemma 12 imply that $\tilde{\mathbb{E}} \succeq 0$ if and only if

$$(P - \mathbb{J}_n/k) \otimes (\mathbb{1}_k - \mathbb{J}_k/k) \succeq 0\,.$$

Since $\mathbb{1}_k - \mathbb{J}_k/k$ is a projection operator, this in turn occurs if and only if

$$P - \mathbb{J}_n/k \succeq 0\,.$$

To summarize, finding a pseudoexpectation is equivalent to finding a PSD matrix $P \in \mathbb{R}^{n\times n}$ with $P_{ii} = 1$ for all i, such that P remains PSD when we subtract the rank-one matrix $\mathbb{J}_n/k$. However, we have thus far only reasoned about the boolean and single color constraints, and including either the coloring or cut constraint places an additional restriction on P. In the case of coloring, we demanded that

$$\sum_c \mathcal{E}_{(i,c),(j,c)} = \sum_c \tilde{\mathbb{E}}[x_{i,c}\, x_{j,c}] = 0 \tag{42}$$

for every edge (i,j). This implies that $\operatorname{tr}\mathcal{E}_{ij} = 0$, and so $P_{ij} = 0$ for each edge. Collecting these observations, a pseudoexpectation for coloring exists exactly when $k > \vartheta(\overline{G})$, where

$$\begin{aligned} \vartheta(\overline{G}) \triangleq \min_P \kappa > 0 \qquad \text{such that} \qquad & P - \mathbb{J}_n/\kappa \succeq 0 \\ & P_{ii} = 1 \qquad \forall i \\ & P_{ij} = 0 \qquad \forall (i,j) \in E\,. \end{aligned} \tag{43}$$

Finally, note that $\mathbb{J}_n/\kappa = v \otimes v$ where $v = \mathbf{1}_n/\sqrt{\kappa}$. Applying Lemma 12 again then gives exactly the PSD (15) for the Lovasz ϑ function, thus completing the proof of Theorem 3.

In the case of good partitions, we required that

$$\sum_{i,j} A_{ij} \sum_c \mathcal{E}_{(i,c),(j,c)} = \sum_{i,j} A_{ij} \sum_c \tilde{\mathbb{E}}[x_{i,c}\, x_{j,c}] = (\tau/k)dn\,, \tag{44}$$

but this means that

$$\sum_{i,j} A_{ij} \operatorname{tr}\mathcal{E}_{ij} = \sum_{i,j} A_{ij} P_{ij} = \langle P, A\rangle = (\tau/k)dn\,.$$

Following the path above, a degree-two pseudoexpectation exists for community detection when $k > \hat{\vartheta}_\tau(\overline{G})$, where

$$\begin{aligned} \hat{\vartheta}_\tau(\overline{G}) \triangleq \min_{P_\tau} \kappa_\tau \qquad \text{such that} \qquad & P_\tau - \mathbb{J}_n/\kappa_\tau \succeq 0 \\ & (P_\tau)_{ii} = 1 \qquad \forall i \\ & \langle P_\tau, A\rangle = (\tau/\kappa_\tau)dn\,. \end{aligned} \tag{45}$$

A priori, it seems that we may need to solve a different SDP for each value of τ, but a bit more work shows that this is not the case. Lemma 12 lets us transform the SDP (17) for $\hat{\vartheta}$ to the following problem,

$$\begin{aligned} \hat{\vartheta}(\overline{G}) \triangleq \min_P \kappa \qquad \text{such that} \qquad & P - \mathbb{J}_n/\kappa \succeq 0 \\ & P_{ii} = 1 \qquad \forall i \\ & \langle P, A\rangle = 0\,. \end{aligned} \tag{46}$$

The following lemma then shows us how to relate optima of (46) to those of (45) for any τ in the disassortative range $\tau < 1$, thus completing the proof of Theorem 4.

▶ **Lemma 13.** *For any $\tau < 1$,*

$$\hat{\vartheta}(\overline{G}) = \frac{\hat{\vartheta}_\tau(\overline{G}) - \tau}{1-\tau}\,. \tag{47}$$

Proof. We show how to translate back and forth between solutions of (45) and (46). Given a matrix P, define

$$P_\tau = (1-\tau/\kappa_\tau)P + (\tau/\kappa_\tau)\mathbb{J}_n \, .$$

It is easy to check that $P_{ii} = 1$ if and only if $(P_\tau)_{ii} = 1$, and $\langle P_\tau, A\rangle = (\tau/\kappa_\tau)dn$ if and only if $\langle P, A\rangle = 0$. Finally, if we set

$$\kappa = \frac{\kappa_\tau - \tau}{1-\tau} \, , \tag{48}$$

then

$$P_\tau - \mathbb{J}_n/\kappa_\tau = (1-\tau/\kappa_\tau)\left(P - \mathbb{J}_n/\kappa\right) \, ,$$

so $P_\tau - \mathbb{J}_n/\kappa_\tau \succeq 0$ if and only if $P - \mathbb{J}_n/\kappa \succeq 0$. Thus (46) is feasible for κ if and only if (45) is feasible for κ_τ. Since $\hat\vartheta(\overline{G})$ and $\hat\vartheta_\tau(\overline{G})$ are the smallest κ and κ_τ respectively for which this is the case, (48) implies (47). ◀

B Proof of Lemma 11

It is immediate that there is such a quadrature rule for polynomials of degree strictly less than t, since the space of linear functionals on such polynomials has dimension t and is thus spanned by the t linearly independent functionals which evaluate at the roots x_i. Now let $\deg u < 2t$. We can divide u by p_t to write $u(z) = a(z)p_t + b(z)$ where $\deg a, \deg b < t$. We have

$$\int_I u(z)\,\mathrm{d}\rho = \int_I \big(a(z)p_t(z) + b(z)\big)\,\mathrm{d}\rho = \langle p_t, a\rangle + \int_I b(z)\,\mathrm{d}\rho = 0 + \sum_{i=1}^t \omega_i b(r_i) = \sum_{i=1}^t \omega_i u(r_i),$$

since p_t is orthogonal to all polynomials of degree less than t and has roots r_i. This verifies exactness of the quadrature rule for polynomials of degree smaller than $2t$.

To show that the weights $\{\omega_i\}$ are positive, let $i \in \{1,\ldots,t\}$ and let $v_i(z) = (p_t(z)/(z - r_i))^2$ be the polynomial with double roots at every root of p_t save r_i. Since v_i is everywhere nonnegative and is a polynomial of degree $2t-2 < t$, we have

$$0 < \int_I v_i(z)\,\mathrm{d}\rho = \sum_{j=1}^t \omega_j v_i(r_j) = \omega_i v_i(r_i) \, ,$$

but since $v(z)$ is nonnegative, ω_i must be positive.

C Proof of Lemma 9

The proof closely follows [7, Theorem 4] which shows that the chromatic number of $G_{n,d}$ is concentrated on two adjacent integers, and which is in turn based on the proof in [42] of two-point concentration for $G(n,p)$ with $p = O(n^{-5/6-\epsilon})$. Recall the configuration model [59], where we make d "copies" of each vertex corresponding to its half-edges, and then choose uniformly from all $(dn-1)!! = (dn)!/(2^{dn/2}(dn/2)!)$ perfect matchings of these copies. If we denote the set of such matchings by $\mathcal{P}_{n,d}$ and condition the corresponding multigraphs on having no self-loops or multiple edges, the resulting distribution is uniform on the set of d-regular graphs, and occupies a constant fraction of the total probability of $\mathcal{P}_{n,d}$. Thus any property which holds with high probability for $\mathcal{P}_{n,d}$ holds with high probability for $G_{n,d}$ as well.

If P, P' are two perfect matchings in $\mathcal{P}_{n,d}$, we write $P \sim P'$ if they differ by a single swap, changing $\{(a,b),(c,d)\}$ to $\{(a,c),(b,d)\}$ or $\{(a,d),(b,c)\}$. The following martingale inequality [59, Theorem 2.19] shows that a random variable which is Lipschitz with respect to these swaps is concentrated.

▶ **Lemma 14.** *Let c be a constant, and let X be a random variable defined on $\mathcal{P}_{n,d}$ such that $|X(P) - X(P')| \le c$ whenever $P \sim P'$. Then*

$$\Pr[|X - \mathbb{E}[X]| > t] \le 2\mathrm{e}^{-\frac{t^2}{dnc}} .$$

Now fix θ, and define X as the minimum number of edge constraints $P_{ij} = 0$ in the SDP (15) violated by an otherwise feasible solution with $\kappa = \theta$. This meets the Lipschitz condition with $c = 2$. By assumption $X = 0$ with positive probability. Lemma 14 then implies that (say) $\mathbb{E}[X] \le (1/2)\sqrt{n \log n}$, in which case $X < \sqrt{n \log n}$ with high probability.

Let S denote the set of endpoints of the violated edges. Then there is an orthogonal representation $\{u_i\}$ of the subgraph induced by $V \setminus S$ and a unit vector $\mathfrak{z}$ such that $\langle u_i, \mathfrak{z} \rangle = 1/\sqrt{\theta}$ and $\langle u_i, u_j \rangle = 0$ if $(i,j) \in E$ and $i, j \notin S$. Our goal is to “fix” $\{u_i\}$ on the violated edges, and if necessary on some additional vertices, to give an orthogonal representation $\{v_i\}$ for all of G.

As in [7, 42], we inductively build a set of vertices $S = U_0, U_1, \ldots, U_T = U$ as follows. Given U_t, let $U_{t+1} = U_t \cup \{i,j\}$ where $i, j \notin U_t$, $(i,j) \in E$, and i and j each have at least one neighbor in U_t. We define T as the step at which there is no such pair i, j and this process ends. Let I denote U’s neighborhood, i.e., the set of vertices outside U which have a neighbor in U. Then I is an independent set, since otherwise the process would have continued. We make the following claim:

▶ **Lemma 15.** *With high probability, the subgraph induced by U is 3-colorable.*

Proof. For all $0 \le t \le T$ we have $|U_t| = 2t + |S|$. Moreover, the subgraph induced by U_t has at least $3t + |S|/2 = (3/2)|U_t| - |S|$ edges and thus average degree at least $3 - 2|S|/|U_t|$. On the other hand, a crude union bound shows that for any d and any $\beta > 2$, there is an $\alpha > 0$ such that, with high probability, all induced subgraphs of G containing αn or fewer vertices have average degree less than β. Since $|S| = o(n)$ with high probability, this implies that $|U_t| \le (2 + o(1))|S|$ for all t, and in particular that $|U| = o(n)$.

The same union bound then implies that with high probability the subgraph induced by $|U|$, and all its subgraphs, have average degree less than 3. But this means that this subgraph has no 3-core: that is, it has at least one vertex of degree less than 3, and so will the subgraph we get by deleting this vertex, and so on. Working backwards, we can 3-color the entire subgraph by starting with the empty set and adding these vertices back in, since at least one of the three colors will always be available to them. ◀

To define our orthogonal representation, let w be a unit vector such that $\langle \mathfrak{z}, w \rangle = \langle u_i, w \rangle = 0$ for all $i \notin S$; such a vector exists since $|S| \ge 2$. Then define

$$\mathfrak{z}' = \sqrt{\frac{\theta}{\theta + 1}}\, \mathfrak{z} + \frac{1}{\sqrt{\theta + 1}}\, w .$$

Then $|\mathfrak{z}'|^2 = 1$, and $\langle w, \mathfrak{z}' \rangle = \langle u_i, \mathfrak{z}' \rangle = 1/\sqrt{\theta + 1}$ for all $i \notin S$. Moreover, there exist three mutually orthogonal unit vectors y_1, y_2, y_3 such that $\langle y_j, \mathfrak{z}' \rangle = 1/\sqrt{\theta + 1}$ and $\langle y_j, w \rangle = 0$ for all $j \in \{1, 2, 3\}$. This follows from the fact that the following matrix is PSD whenever $\theta \geq 3$, in which case it can be realized as the Gram matrix of $\{y_1, y_2, y_3, w, \mathfrak{z}'\}$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \frac{1}{\sqrt{\theta+1}} \\ 0 & 1 & 0 & 0 & \frac{1}{\sqrt{\theta+1}} \\ 0 & 0 & 1 & 0 & \frac{1}{\sqrt{\theta+1}} \\ 0 & 0 & 0 & 1 & \frac{1}{\sqrt{\theta+1}} \\ \frac{1}{\sqrt{\theta+1}} & \frac{1}{\sqrt{\theta+1}} & \frac{1}{\sqrt{\theta+1}} & \frac{1}{\sqrt{\theta+1}} & 1 \end{pmatrix} .$$

Finally, let $\sigma(i) \in \{1, 2, 3\}$ be a proper 3-coloring of the subgraph induced by U. Then the following is an orthogonal representation of G,

$$v_i = \begin{cases} u_i & i \in V \setminus (U \cup I) \\ w & i \in I \\ y_{\sigma(i)} & i \in U \,, \end{cases}$$

and $\langle v_i, \mathfrak{z}' \rangle = 1/\sqrt{\theta + 1}$ for all i. This gives a feasible solution to the SDP (15) with $\kappa = \theta + 1$, implying that $\vartheta(\overline{G}) \leq \theta + 1$.

Cutoff for a Stratified Random Walk on the Hypercube

Anna Ben-Hamou[1] and Yuval Peres[2]

1 **IMPA, Rio de Janeiro, Brasil**
`benhamou@impa.br`

2 **Microsoft Research, Redmond, WA, USA**
`peres@microsoft.com`

Abstract

We consider the random walk on the hypercube which moves by picking an ordered pair (i, j) of distinct coordinates uniformly at random and adding the bit at location i to the bit at location j, modulo 2. We show that this Markov chain has cutoff at time $\frac{3}{2}n \log n$ with window of size n, solving a question posed by Chung and Graham (1997).

1998 ACM Subject Classification G.3 Probability

Keywords and phrases Mixing times, cutoff, hypercube

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.29

1 Introduction

Let $\mathrm{SL}_n(\mathbb{Z}_2)$ be the set of invertible matrices with coefficients in $\mathbb{Z}_2$, and consider the Markov chain on $\mathrm{SL}_n(\mathbb{Z}_2)$ which moves by picking two distinct rows at random and adding the first one to the other. This walk has received significant attention, both from group theoreticians and cryptologists. Diaconis and Saloff-Coste [4] showed that the ℓ_2-mixing time was $O(n^4)$, and the powerful results of Kassabov [5] yield the upper-bound $O(n^3)$. One may observe that if $Z_t \in \{0, 1\}^n \backslash \{\mathbf{0}\}$ denotes the first column of the matrix at time t, then the process $\{Z_t\}_{t\geq 0}$ is also a Markov chain (defined more precisely below). Diaconis and Saloff-Coste [4] showed that the log-Sobolev constant of this chain is $O(n^2)$, which yields an upper-bound of order $n^2 \log n$ on the ℓ_2-mixing time. They however conjectured that the right order for the total-variation mixing was $n \log n$. Chung and Graham [3] confirmed this conjecture. They showed that the relaxation time of $\{Z_t\}$ was of order n (which yields a tight upper-bound of order n^2 for ℓ_2-mixing) and that the total-variation mixing time $t_{\mathrm{mix}}(\varepsilon)$ was smaller than $c_\varepsilon n \log n$ for some constant c_ε. They asked whether one could make this bound more precise and replace c_ε by a universal constant which would not depend on ε. We answer this question positively by proving that the chain $\{Z_t\}$ has cutoff at time $\frac{3}{2}n \log n$, with window of order n.

The matrix walk problem was brought to our attention by Ron Rivest, who was mostly interested in computational mixing aspects [7]. The question of determining the total-variation mixing time of this walk is still largely open. By a diameter bound, it can be lower bounded by $\Omega\left(\frac{n^2}{\log n}\right)$ (see Andrén et al. [1], Christofides [2]). The best known upper-bound is $O(n^3)$ as established by Kassabov [5].

Main result

Let $\mathcal{X} = \{0, 1\}^n \backslash \{\mathbf{0}\}$ and consider the Markov chain $\{Z_t\}_{t\geq 0}$ on $\mathcal{X}$ defined as follows: if the current state is x and if $x(i)$ denotes the bit at the i^{th} coordinate of x, then the walk

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 29; pp. 29:1–29:10
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

proceeds by choosing uniformly at random an ordered pair (i, j) of distinct coordinates, and replacing $x(j)$ by $x(j) + x(i) \pmod 2$.

The transition matrix P of this chain is symmetric, irreducible and aperiodic. Its stationary distribution π is the uniform distribution over $\mathcal{X}$, i.e. for all $x \in \mathcal{X}$, $\pi(x) = \frac{1}{2^n-1}$. We are interested in the total-variation mixing time, defined as

$$t_{\text{mix}}(\varepsilon) = \min\{t \geq 0,\, d(t) \leq \varepsilon\}\ ,$$

where $d(t) = \max\limits_{x \in \mathcal{X}} d_x(t)$ and $d_x(t)$ is the total-variation distance between $P^t(x, \cdot)$ and π:

$$d_x(t) = \sup_{A \subset \mathcal{X}} \big(\pi(A) - P^t(x, A)\big) \;=\; \sum_{y \in \mathcal{X}} \big(P^t(x, y) - \pi(y)\big)_+ \ .$$

▶ **Theorem 1.** *The chain $\{Z_t\}$ has total-variation cutoff at time $\frac{3}{2} n \log n$ with window n, i.e.*

$$\lim_{\alpha \to +\infty} \liminf_{n \to +\infty} d\left(\frac{3}{2} n \log n - \alpha n\right) = 1\,,$$

and

$$\lim_{\alpha \to +\infty} \limsup_{n \to +\infty} d\left(\frac{3}{2} n \log n + \alpha n\right) = 0\,.$$

Before proving Theorem 1, we first state some useful properties of the birth-and-death chain given by the Hamming weight of Z_t. In particular, we show that this projected chain also has cutoff at $\frac{3}{2} n \log n$ (Section 2). Section 3 is then devoted to the proof of Theorem 1.

2 The Hamming weight

For a vertex $x \in \mathcal{X}$, we denote by $H(x)$ the Hamming weight of x, *i.e.*

$$H(x) = \sum_{i=1}^{n} x(i)\,.$$

Consider the birth-and-death chain $H_t := H(Z_t)$, and denote by P_H, π_H, and $d_H(\cdot)$ its transition matrix, stationary distribution, and total-variation distance to equilibrium. For $1 \leq k \leq n$, we have

$$\begin{aligned} P_H(k, k+1) &= \frac{k(n-k)}{n(n-1)}\,, \\ P_H(k, k-1) &= \frac{k(k-1)}{n(n-1)}\,, \\ P_H(k, k) &= \frac{n-k}{n}\,, \end{aligned}$$

and

$$\pi_H(k) = \frac{\binom{n}{k}}{2^n - 1}\,.$$

The hitting time of state k is defined as

$$T_k = \min\{t \geq 0,\, H_t = k\}\ .$$

One standard result in birth-and-death chains is that, for $2 \le \ell \le n$,

$$\mathbb{E}_{\ell-1}(T_\ell) = \frac{1}{P_H(\ell, \ell-1)} \sum_{i=1}^{\ell-1} \frac{\pi_H(i)}{\pi_H(\ell)}\,, \tag{1}$$

(see for instance [6, Section 2.5]). The following lemma will be useful.

▶ **Lemma 2.** *Let $0 < \beta < 1$ and $K = (1-\beta)\frac{n}{2}$. Then there exist constants $c_\beta, C_\beta \in \mathbb{R}$ depending on β only such that*

$$\mathbb{E}_1(T_K) \le n \log n + c_\beta n\,,$$

and

$$\mathrm{Var}_1 T_K \le C_\beta n^2\,.$$

Proof of Lemma 2. For $2 \le k \le K$, let $\mu_k = \mathbb{E}_{k-1} T_k$ and $v_k = \mathrm{Var}_{k-1}(T_k)$. Resorting to (1), we have

$$\mu_k = \frac{\binom{n}{k-1}}{\binom{n-2}{k-2}} \sum_{i=1}^{k-1} \frac{\binom{n}{i}}{\binom{n}{k-1}} \le \frac{\binom{n}{k-1}}{\binom{n-2}{k-2}} \sum_{i=1}^{k-1} \left(\frac{k-1}{n-k+2}\right)^{k-i-1} \le \frac{n^2}{k(n-2k)} \tag{2}$$

Summing from 2 to K yields the desired bound on $\mathbb{E}_1 T_K$. Moving on to the variance, by independence of the successive hitting times, we have

$$\mathrm{Var}_1 T_K = \sum_{k=1}^{K-1} v_{k+1}\,.$$

Hence, it is sufficient to show that there exists a constant $a_\beta > 0$ such that $v_{k+1} \le \frac{a_\beta n^2}{k^2}$ for all $k \le K$. To do so, we consider the following distributional identity for the hitting time T_{k+1} starting from k:

$$T_{k+1} = 1 + (1-I)\widetilde{T}_{k+1} + IJ(\widehat{T}_k + \widehat{T}_{k+1})\,,$$

where I is the indicator that the chain moves (*i.e.* that a one is picked as updating coordinate), J is the indicator that the chain decreases given that it moves (*i.e.* that the chosen one is added to another one), $\widetilde{T}_{k+1}$ and $\widehat{T}_{k+1}$ are copies of T_{k+1}, and $\widehat{T}_k$ is the hitting time of k starting from $k-1$. All those variables may be assumed to be independent. After computation we obtain the following induction relation:

$$\begin{aligned} v_{k+1} &= \frac{k-1}{n-1}(v_k + v_{k+1}) + \left(1 - \frac{k}{n}\right)\mu_{k+1}^2 + \frac{k-1}{n-1}\left(1 - \frac{k(k-1)}{n(n-1)}\right)(\mu_k + \mu_{k+1})^2 \\ &\le \frac{k}{n}(v_k + v_{k+1}) + \mu_{k+1}^2 + \frac{k}{n}(\mu_k + \mu_{k+1})^2\,. \end{aligned}$$

Using the fact that for all $k \le K$, we have $\mu_k \le \frac{n}{\beta k}$ (which can be seen by inequality (2)), and after some simplification, we get

$$v_{k+1} \le \frac{k}{n-k} v_k + \frac{3n^3}{\beta^2 k^2 (n-k)} \le \frac{k}{n-k} v_k + \frac{6n^2}{\beta^2 k^2}\,.$$

By induction and using that $v_2 \le n^2$, we obtain that $v_{k+1} \le \frac{a_\beta n^2}{k^2}$ for all $k \le K$. ◀

The following proposition establishes cutoff for the chain $\{H_t\}$ and will be used in the next section to prove cutoff for the chain $\{Z_t\}$.

▶ **Proposition 3.** *The chain H_t exhibits cutoff at time $\frac{3}{2}n\log n$ with window n.*

Proof. For the lower bound, we want to show that for $t = \frac{3}{2}n\log n - 2\alpha n$

$$d_H(t) \geq 1 - \varepsilon(\alpha)\,,$$

where $\varepsilon(\alpha) \to 0$ as $\alpha \to +\infty$. Consider the chain started at $H_0 = 1$ and let $k = \frac{n}{2} - \alpha\sqrt{n}$ and $A = \{k, k+1, \ldots, n\}$. By definition of total-variation distance,

$$d_H(t) \geq \pi_H(A) - P_H^t(1, A) \;\geq\; \pi_H(A) - \mathbb{P}_1(T_k \leq t)\,.$$

By the Central Limit Theorem, $\lim\limits_{\alpha\to\infty}\lim\limits_{n\to\infty} \pi_H(A) = 1$. Moving on to $\mathbb{P}_1(T_k \leq t)$, let us write

$$\mathbb{P}_1(T_k \leq t) = \mathbb{P}_1\left(T_{n/3} \leq n\log n - \alpha n\right) + \mathbb{P}_{n/3}\left(T_k \leq \frac{n\log n}{2} - \alpha n\right)\,.$$

Note that $T_{n/3}$ is stochastically larger than $\sum_{i=1}^{n/3} G_i$, where $(G_i)_{i=1}^{n/3}$ are independent Geometric random variables with respective parameter i/n (this is because at each step, we need at least to pick a one to just move from the current position). By Chebyshev's Inequality,

$$\mathbb{P}_1\left(T_{n/3} \leq n\log n - \alpha n\right) = O\left(\frac{1}{\alpha^2}\right)\,.$$

Now, starting from Hamming weight $n/3$ and up to time T_k, we may couple H_t with $\widetilde{H}_t$, the Hamming weight of the standard lazy random walk on the hypercube (at each step, pick a coordinate uniformly at random and randomize the bit at this coordinate), in such a way $T_k \geq S_k$, where $S_k = \inf\{t \geq 0, \widetilde{H}_t = k\}$. It is known that S_k satisfies

$$\mathbb{P}_{n/3}\left(S_k \leq \frac{n\log n}{2} - \alpha n\right) \leq \varepsilon(\alpha)\,,$$

with $\varepsilon(\alpha) \to 0$ as $\alpha \to +\infty$ (see for instance the proof of [6, Proposition 7.13]), which concludes the proof of the lower bound.

For the upper bound, letting $t = \frac{3}{2}n\log n + 2\alpha n$, we have

$$d_H(t) \leq \mathbb{P}_1\left(T_{n/3} > n\log n + \alpha n\right) + \max_{k\geq n/3} d_H^{(k)}\left(\frac{n\log n}{2} + \alpha n\right)\,. \tag{3}$$

Lemma 2 entails that $T_{n/3}$ concentrates well: $\mathbb{E}_1(T_{n/3}) = n\log n + cn$ for some absolute constant c, and $\mathrm{Var}_1(T_{n/3}) = O(n^2)$. By Chebyshev's Inequality,

$$\mathbb{P}_1\left(T_{n/3} > n\log n + \alpha n\right) = O\left(\frac{1}{\alpha^2}\right)\,. \tag{4}$$

To control the second term in the right-hand side of (3), we use the coupling method (see Levin et al. [6, Chapter 5]). For all starting point $k \geq n/3$, we consider the following coupling between a chain H_t started at k and a chain H_t^π started from stationarity: at each step t, if H_t makes an actual move (a one is picked as updating bit in the underlying chain Z_t), we try "as much as possible" not to move H_t^π (picking a zero as updating bit). Conversely, when H_t does not move, we try "as much as possible" to move H_t^π, the goal being to increase the chance that the two chains do not cross each other. The chains stay together once they have met for the first time. We claim that the study of the coupling time can be reduced to the study of the first time when the chain started at $n/3$ reaches $n/2$. Indeed, as $\pi_H([2n/3,n]) = o(1)$, with high probability, $H_0^\pi \le 2n/3$, and as starting from a larger Hamming weight can only speed up the chain, $\mathbb{P}_{2n/3}(T_{n/2} > t) \le \mathbb{P}_{n/3}(T_{n/2} > t)$. Now, when both chains have reached $n/2$, either they have met, or they have crossed each other. In this last situation, we know however that the expected time of their first return to $n/2$ is $O(\sqrt{n})$, so that $\mathbb{P}_{n/2}\left(T_{n/2}^+ > \sqrt{\alpha n}\right) = O(1/\sqrt{\alpha})$. Moreover, thanks to our coupling, during each of those excursions, the chains have positive probability to meet, so that after an additional time of order $\alpha\sqrt{n}$ we can guarantee that they have met with large probability. We are thus left to prove that $\mathbb{P}_{n/3}\left(T_{n/2} > \frac{n\log n}{2} + \alpha n\right) \le \varepsilon(\alpha)$, for a function ε tending to 0 at $+\infty$.

Starting from $H_0 = n/3$, we first argue that H_t will remain above $2n/7$ for a very long time. Namely, defining $\mathcal{G}_t = \left\{T_{2n/7} > t\right\}$, we have

$$\mathbb{P}_{n/3}\left(\mathcal{G}_{n^2}\right) = 1 - o(1)\,. \tag{5}$$

This can easily be seen by considering $T_k^+ = \min\{t \ge 1,\, H_t = k\}$ and taking a union bound over the excursions around $k = n/3$ which visit $m = 2n/7$:

$$\mathbb{P}_k(T_m \le n^2) \le n^2 \mathbb{P}_k(T_m \le T_k^+)\,,$$

and

$$\mathbb{P}_k(T_m \le T_k^+) = \frac{\mathbb{E}_k(T_k^+)}{\mathbb{E}_m(T_k) + \mathbb{E}_k(T_m)} \le \frac{\mathbb{E}_k(T_k^+)}{\mathbb{E}_m(T_m^+)} = \frac{\pi_H(m)}{\pi_H(k)}\,,$$

which decreases exponentially fast in n.

Our goal now will be to analyse the tail of $\tau = \inf\{t \ge 0,\, D_t \le 0\}$, where

$$D_t = \frac{n}{2} - H_t\,.$$

Observe that

$$D_{t+1} - D_t = \begin{cases} 1 & \text{with probability } \frac{H_t(H_t-1)}{n(n-1)} \\ -1 & \text{with probability } \frac{H_t(n-H_t)}{n(n-1)} \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

We get

$$\mathbb{E}\left[D_{t+1} - D_t \,\middle|\, D_t\right] = -\frac{2\left(\frac{n}{2} - D_t\right)(D_t + 1)}{n(n-1)} \le -\frac{D_t}{n} + \frac{2D_t^2}{n(n-1)}\,. \tag{7}$$

Writing a similar recursion for the second moment of D_t gives

$$\mathbb{E}\left[D_{t+1}^2 - D_t^2 \,\middle|\, D_t\right] = -\frac{4H_tD_t(D_t + 1/2)}{n(n-1)} + \frac{H_t}{n} \le -\frac{4H_tD_t^2}{n^2} + 2\,.$$

On the event $\mathcal{G}_t$,

$$\mathbb{E}\left[D_{t+1}^2 - D_t^2 \,\middle|\, D_t\right] \le -\frac{8D_t^2}{7n} + 2\,.$$

By induction, letting $\mathcal{D}_t = \mathbb{1}_{\mathcal{G}_t} D_t$ (and noticing that $\mathcal{G}_{t+1} \subset \mathcal{G}_t$), we get

$$\mathbb{E}\left[\mathcal{D}_t^2\right] \le \mathbb{E}[D_0^2]\left(1-\frac{8}{7n}\right)^t + \frac{7n}{4} \;\le\; \frac{n^2}{4}\,\mathrm{e}^{-8t/7n} + 2n\,.$$

Plugging this back in (7),

$$\mathbb{E}\left[\mathcal{D}_{t+1}\right] \le \left(1-\frac{1}{n}\right)\mathbb{E}\left[\mathcal{D}_t\right] + \,\mathrm{e}^{-8t/7n} + 4/n\,,$$

and by induction,

$$\mathbb{E}\left[\mathcal{D}_t\right] \le an\,\mathrm{e}^{-t/n} + b\,, \tag{8}$$

for absolute constants $a, b \ge 0$. Also, letting $\tau_\star = \inf\{t \ge 0,\, \mathcal{D}_t = 0\}$, we see by (6) that, provided $\tau_\star > t$, the process $\{\mathcal{D}_t\}$ is at least as likely to move downwards than to move upwards and that there exists a constant $\sigma^2 > 0$ such that $\operatorname{Var}\left(\mathcal{D}_{t+1}\,\middle|\,\mathcal{D}_t\right) \ge \sigma^2$ (this is because, on $\mathcal{G}_t$ the probability to make a move at time t in larger than some positive absolute constant). By Levin et al. [6, Proposition 17.20], we know that for all $u > 0$ and $k \ge 0$,

$$\mathbb{P}_k(\tau_\star > u) \le \frac{4k}{\sigma\sqrt{u}}\,. \tag{9}$$

Now take $H_0 = n/3$, $D_0 = n/6$, $s = \frac{1}{2}n\log n$ and $u = \alpha n$. We have

$$\mathbb{P}_{D_0}\left(\tau > s+u\right) \le \mathbb{P}_{\mathcal{D}_0}\left(\tau_\star > s+u\right) + \mathbb{P}_{H_0}\left(\mathcal{G}^c_{n^2}\right)\,.$$

By equation (5), $\mathbb{P}_{H_0}\left(\mathcal{G}^c_{n^2}\right) = o(1)$, and, combining (9) and (8), we have

$$\mathbb{P}_{\mathcal{D}_0}\left(\tau_\star > s+u\right) = \mathbb{E}_{\mathcal{D}_0}\left[\mathbb{P}_{\mathcal{D}_s}\left(\tau_\star > u\right)\right] \;\le\; \mathbb{E}_{\mathcal{D}_0}\left[\frac{4\mathcal{D}_s}{\sigma\sqrt{u}}\right] \;=\; O\left(\frac{1}{\sqrt{\alpha}}\right)\,,$$

which implies

$$\max_{k \ge n/3} d_H^{(k)}\left(s+u\right) = O\left(\frac{1}{\sqrt{\alpha}}\right)\,, \tag{10}$$

and concludes the proof of the upper bound. ◀

3 Proof of Theorem 1

First note that, as projections of chains can not increase total-variation distance, the lower bound on $d(t)$ readily follows from the lower bound on $d_H(t)$, as established in Proposition 3. Therefore, we only have to prove the upper bound.

Let $\mathcal{E} = \{x \in \mathcal{X},\, H(x) \ge n/3\}$ and $\tau_{\mathcal{E}}$ be the hitting time of set $\mathcal{E}$. For all $t, s > 0$, we have

$$d(t+s) \le \max_{x_0 \in \mathcal{X}} \mathbb{P}_{x_0}\left(\tau_{\mathcal{E}} > s\right) + \max_{x \in \mathcal{E}} d_x(t)\,.$$

By (4), taking $s = n\log n + \alpha n$, we have $\max_{x_0 \in \mathcal{X}} \mathbb{P}_{x_0}(\tau_{\mathcal{E}} > s) = O(1/\alpha^2)$, so that our task comes down to showing that for all $x \in \mathcal{E}$,

$$d_x\left(\frac{n\log n}{2} + \alpha n\right) \le \varepsilon(\alpha)\,,$$

with $\varepsilon(\alpha) \to 0$ as $\alpha \to +\infty$. Let us fix $x \in \mathcal{E}$. Without loss of generality, we may assume that x is the vertex with $\bar{x} \geq n/3$ ones on the first $\bar{x}$ coordinates, and $n - \bar{x}$ zeros on the last $n - \bar{x}$ coordinates. We denote by $\{Z_t\}$ the random walk started at $Z_0 = x$ and for a vertex $z \in \mathcal{X}$, we define a two-dimensional object $\mathbf{W}(z)$, keeping track of the number of ones within the first $\bar{x}$ and last $n - \bar{x}$ coordinates of z, that is

$$\mathbf{W}(z) = \left(\sum_{i=1}^{\bar{x}} z(i), \sum_{i=\bar{x}+1}^{n} z(i) \right) .$$

The projection of $\{Z_t\}_{t\geq 0}$ induced by $\mathbf{W}$ will be denoted $\mathbf{W}_t = \mathbf{W}(Z_t) = (X_t, Y_t)$. We argue that the study of $\{Z_t\}_{t\geq 0}$ can be reduced to the study of $\{\mathbf{W}_t\}_{t\geq 0}$, and that, when coupling two chains distributed as $\mathbf{W}_t$, we can restrict ourselves to initial states with the same total Hamming weight. Indeed, letting $\nu_{\bar{x}}$ be the uniform distribution over $\{z \in \mathcal{X},\, H(z) = \bar{x}\}$. By the triangle inequality,

$$d_x(t) \leq \left\| \mathbb{P}_x\left(Z_t \in \cdot\right) - \mathbb{P}_{\nu_{\bar{x}}}\left(Z_t \in \cdot\right) \right\|_{\mathrm{TV}} + \left\| \mathbb{P}_{\nu_{\bar{x}}}\left(Z_t \in \cdot\right) - \pi(\cdot) \right\|_{\mathrm{TV}} \tag{11}$$

Starting from $\nu_{\bar{x}}$, the conditional distribution of Z_t given $\{H(Z_t) = h\}$ is uniform over $\{y \in \mathcal{X},\, H(y) = h\}$. This entails

$$\left\| \mathbb{P}_{\nu_{\bar{x}}}\left(Z_t \in \cdot\right) - \pi(\cdot) \right\|_{\mathrm{TV}} = \left\| \mathbb{P}_{\bar{x}}\left(H_t \in \cdot\right) - \pi_H(\cdot) \right\|_{\mathrm{TV}} .$$

For $t = \frac{n \log n}{2} + \alpha n$, we know by (10) in the proof of Proposition 3 that $\left\| \mathbb{P}_{\bar{x}}\left(H_t \in \cdot\right) - \pi_H(\cdot) \right\|_{\mathrm{TV}} = O(1/\sqrt{\alpha})$. As for the first term in the right-hand side of (11), note that if z and z' are two vertices such that $\mathbf{W}(z) = \mathbf{W}(z')$, then for all $t \geq 0$, $\mathbb{P}_x(Z_t = z) = \mathbb{P}_x(Z_t = z')$, and that for all $y \in \mathcal{X}$ such that $\mathbf{W}(y) = (k, \ell)$

$$\begin{aligned}
\mathbb{P}_{\nu_{\bar{x}}}\left(Z_t = y\right) &= \sum_{\substack{i,j \\ i+j=\bar{x}}} \sum_{z,\, \mathbf{W}(z)=(i,j)} \frac{1}{\binom{n}{\bar{x}}} \mathbb{P}_z\left(Z_t = y\right) \\
&= \sum_{\substack{i,j \\ i+j=\bar{x}}} \frac{\binom{\bar{x}}{i}\binom{n-\bar{x}}{j}}{\binom{n}{\bar{x}}} \sum_{z,\, \mathbf{W}(z)=(i,j)} \frac{\mathbb{P}_z\left(Z_t = y\right)}{\binom{\bar{x}}{i}\binom{n-\bar{x}}{j}} \\
&= \sum_{\substack{i,j \\ i+j=\bar{x}}} \frac{\binom{\bar{x}}{i}\binom{n-\bar{x}}{j}}{\binom{n}{\bar{x}}} \frac{\mathbb{P}_{(i,j)}\left(\mathbf{W}_t = (k, \ell)\right)}{\binom{\bar{x}}{k}\binom{n-\bar{x}}{\ell}} .
\end{aligned}$$

Hence,

$$\left\| \mathbb{P}_x\left(Z_t \in \cdot\right) - \mathbb{P}_{\nu_{\bar{x}}}\left(Z_t \in \cdot\right) \right\|_{\mathrm{TV}} \leq \max_{\substack{i,j \\ i+j=\bar{x}}} \left\| \mathbb{P}_{(\bar{x},0)}\left(\mathbf{W}_t \in \cdot\right) - \mathbb{P}_{(i,j)}\left(\mathbf{W}_t \in \cdot\right) \right\|_{\mathrm{TV}} .$$

Now let $y \in \mathcal{E}$ such that $H(y) = \bar{x}$, and consider the chains $Z_t, \widetilde{Z}_t$ started at x and y respectively. Let $\mathbf{W}(Z_t) = (X_t, Y_t)$ and $\mathbf{W}(\widetilde{Z}_t) = (\widetilde{X}_t, \widetilde{Y}_t)$. We couple Z_t and $\widetilde{Z}_t$ as follows: at each step t, provided $H(Z_t) = H(\widetilde{Z}_t)$ and $\mathbf{W}(Z_t) \neq \mathbf{W}(\widetilde{Z}_t)$, we consider a random permutation π_t which is such that $Z_t(i) = \widetilde{Z}_t(\pi_t(i))$ for all $1 \leq i \leq n$, that is, we pair uniformly at random the ones (resp. the zeros) of Z_t with the ones (resp. the zeros) of $\widetilde{Z}_t$. If Z_t moves to Z_{t+1} by choosing the pair (i_t, j_t) and updating $Z_t(j_t)$ to $Z_t(j_t) + Z_t(i_t)$, then we move from $\widetilde{Z}_t$ to $\widetilde{Z}_{t+1}$ by updating $\widetilde{Z}_t(\pi_t(j_t))$ to $\widetilde{Z}_t(\pi_t(j_t)) + \widetilde{Z}_t(\pi_t(i_t))$. Once $\mathbf{W}(Z_t) = \mathbf{W}(\widetilde{Z}_t)$, the permutation π_t is chosen in such a way that the ones in the top (resp. in the bottom) in Z_t are matched with the ones in the top (resp. in the bottom) in $\widetilde{Z}_t$, guaranteeing that from that time $\mathbf{W}(Z_t)$ and $\mathbf{W}(\widetilde{Z}_t)$ remain equal. Note that this coupling ensures that for all $t \geq 0$, the Hamming weight of Z_t is equal to that of $\widetilde{Z}_t$, and we may unequivocally denote it by H_t. In particular, coupling of the chains $\mathbf{W}(Z_t)$ and $\mathbf{W}(\widetilde{Z}_t)$ occurs when X_t and $\widetilde{X}_t$ are matched. As $X_t \geq \widetilde{X}_t$ for all $t \geq 0$, we may consider

$$\tau = \inf\{t \geq 0,\, \mathbf{D}_t = 0\}\,,$$

where $\mathbf{D}_t = X_t - \widetilde{X}_t$.

Before analysing the behaviour of $\{\mathbf{D}_t\}$, we first notice that the worst possible y for the coupling time satisfies $\mathbf{W}(y) = (\max\{0, 2\bar{x} - n\}, \min\{\bar{x}, n - \bar{x}\})$. We now fix y to be such a vertex, and show that, starting from x, y, the variables $\mathbf{W}(Z_t), \mathbf{W}(\widetilde{Z}_t)$ remain "nice" for a very long time. More precisely, defining

$$\mathcal{B}_t = \bigcap_{s=0}^{t} \left\{ H_s \geq 2n/7,\, X_s \geq \frac{\bar{x}}{p},\, \widetilde{Y}_s \geq \frac{\min\{\bar{x}, n - \bar{x}\}}{p} \right\}\,,$$

we claim that we can choose $p \geq 1$ fixed such that

$$\mathbb{P}_{x,y}\left(\mathcal{B}_{n^2}\right) = 1 - o(1)\,. \tag{12}$$

Indeed, the fact that $\mathbb{P}_{n/3}(T_{2n/7} \leq n^2) = o(1)$ has already been established in the proof of Proposition 3 (equation (5)), and with the same kind of arguments, we show that $\mathbb{P}_{(\bar{x},0)}\left(\cup_{s=0}^{n^2}\{X_s < \bar{x}/p\}\right) = o(1)$. Letting $A = \{(\bar{x}/p, \ell),\, \ell = 0, \ldots, n - \bar{x}\}$, $\pi_{\mathbf{W}}$ be the stationary distribution of $\mathbf{W}_t$, and $k_{\bar{x}} = \min\left\{\frac{\bar{x}}{2}, \frac{n-\bar{x}}{2}\right\}$, we have

$$\begin{aligned}\mathbb{P}_{(\bar{x},0)}(T_A \leq n^2) \leq \mathbb{P}_{(\bar{x}/2,k_{\bar{x}})}(T_A \leq n^2) \;&\leq\; n^2 \sum_{\ell=0}^{n-\bar{x}} \mathbb{P}_{(\bar{x}/2,k_{\bar{x}})}\left(T_{(\bar{x}/p,\ell)} \leq T^{+}_{(\bar{x}/2,k_{\bar{x}})}\right)\\ &\leq n^2 \sum_{\ell=0}^{n-\bar{x}} \frac{\pi_{\mathbf{W}}(\bar{x}/p,\ell)}{\pi_{\mathbf{W}}(\bar{x}/2,k_{\bar{x}})} \;=\; \frac{n^2 2^{n-\bar{x}}\binom{\bar{x}}{\bar{x}/p}}{\binom{\bar{x}}{\bar{x}/2}\binom{n-\bar{x}}{k_{\bar{x}}}}\,,\end{aligned}$$

and we can choose p large enough such that this quantity decreases exponentially fast in n. Similarly, starting from y, the value of $\widetilde{Y}_s$ will remain at a high level for a very long time, establishing (12).

Let us now turn to the analysis of $\{\mathbf{D}_t\}$. On the event $\{t < \tau\}$,

$$\mathbf{D}_{t+1} - \mathbf{D}_t = \begin{cases} 1 & \text{with probability } p_1^t \\ -1 & \text{with probability } p_{-1}^t \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

where

$$p_1^t = \frac{H_t}{n} \cdot \frac{n - H_t}{n-1} \cdot \frac{\bar{x} - X_t}{n - H_t} \cdot \frac{n - \bar{x} - \widetilde{Y}_t}{n - H_t} + \frac{H_t}{n} \cdot \frac{H_t - 1}{n-1} \cdot \frac{Y_t}{H_t} \cdot \frac{\widetilde{X}_t}{H_t}\,,$$

and

$$p_{-1}^t = \frac{H_t}{n} \cdot \frac{n - H_t}{n-1} \cdot \frac{\bar{x} - \widetilde{X}_t}{n - H_t} \cdot \frac{n - \bar{x} - Y_t}{n - H_t} + \frac{H_t}{n} \cdot \frac{H_t - 1}{n-1} \cdot \frac{X_t}{H_t} \cdot \frac{\widetilde{Y}_t}{H_t}\,.$$

After computation, we get, on $\{t < \tau\}$,

$$\mathbb{E}\left[\mathbf{D}_{t+1} - \mathbf{D}_t \,\middle|\, Z_t, \widetilde{Z}_t\right] = -\frac{H_t \mathbf{D}_t}{n(n-1)}\left(1 + \frac{H_t - 1}{H_t}\right) \;\leq\; -\frac{\mathbf{D}_t}{n^2}\,(2H_t - 1) \tag{14}$$

From (14), it is not hard to see that the variable

$$M_t = \mathbb{1}_{\{\tau>t\}}\mathbf{D}_t \exp\left(\sum_{s=0}^{t-1}\frac{(2H_s-1)}{n^2}\right)$$

is a super-martingale, which implies $\mathbb{E}_{x,y}\left[M_t\right] \le \mathbb{E}_{x,y}\left[\mathbf{D}_0\right] \le n$.

Now let $\tau_\star = \inf\{t \ge 0,\ \mathbb{1}_{\mathcal{B}_t}\mathbf{D}_t = 0\}$. By (13), we see that, provided $\{\tau_\star > t\}$, the process $\{\mathbb{1}_{\mathcal{B}_t}\mathbf{D}_t\}$ is a supermartingale ($p_{-1}^t \ge p_1^t$) and that there exists a constant $\sigma^2 > 0$ such that the conditional variance of its increments is larger than σ^2 (because on $\mathcal{B}_t$, the probability to make a move $p_{-1}^t + p_1^t$ is larger than some absolute constant). By Levin et al. [6, Proposition 17.20], for all $u > 0$ and $k \ge 0$,

$$\mathbb{P}_k(\tau_\star > u) \le \frac{4k}{\sigma\sqrt{u}}\cdot \tag{15}$$

Now take $t = \frac{n\log n}{2}$ and $u = \alpha n$. We have

$$\mathbb{P}_{x,y}(\tau > t+u) \le \mathbb{P}_{x,y}(\mathcal{B}_{n^2}^c) + \mathbb{P}_{x,y}(\tau_\star > t+u)\,.$$

By (12), we know that $\mathbb{P}_{x,y}(\mathcal{B}_{n^2}^c) = o(1)$. Also, considering the event

$$\mathcal{A}_{t-1} = \left\{\sum_{s=0}^{t-1} H_s \ge \frac{n^2\log n}{4} - \beta n^2\right\},$$

and resorting to (15), we get

$$\begin{aligned}\mathbb{P}_{x,y}\left(\tau_\star > t+u\right) &\le \mathbb{E}_{x,y}\left[\mathbb{1}_{\{\tau_\star>t\}}\mathbb{P}_{Z_t,\widetilde{Z}_t}\left(\tau_\star > u\right)\right]\\ &\le \mathbb{P}_{x,y}\left(\{\tau_\star > t\}\cap\mathcal{A}_{t-1}^c\right) + \mathbb{E}_{x,y}\left[\mathbb{1}_{\mathcal{A}_{t-1}}\mathbb{1}_{\{\tau_\star>t\}}\frac{4\mathbf{D}_t}{\sigma\sqrt{u}}\right]\cdot\end{aligned}$$

On the one hand, recalling the notation and results of Section 2 (in particular equation (8)), and applying Markov's Inequality,

$$\begin{aligned}\mathbb{P}_{x,y}\left(\{\tau_\star > t\}\cap\mathcal{A}_{t-1}^c\right) &\le \mathbb{P}_{x,y}\left(\sum_{s=0}^{t-1}\mathcal{D}_s > \beta n^2\right)\\ &\le \frac{1}{\beta n^2}\sum_{s=0}^{t-1}\left(an\,\mathrm{e}^{-s/n} + b\right) \;=\; O\left(\frac{1}{\beta}\right)\cdot\end{aligned}$$

On the other hand,

$$\mathbb{E}_{x,y}\left[\mathbb{1}_{\{\tau_\star>t\}}\mathbb{1}_{\mathcal{A}_{t-1}}\mathbf{D}_t\right] \le \exp\left(-\frac{\log n}{2} + \frac{t}{n^2} + 2\beta\right)\mathbb{E}_{x,y}\left[M_t\right] \;=\; O\left(\mathrm{e}^{2\beta}\sqrt{n}\right)\,.$$

In the end, we get

$$\mathbb{P}_{x,y}\left(\tau > t+u\right) = O\left(\frac{1}{\beta} + \frac{\mathrm{e}^{2\beta}}{\sqrt{\alpha}}\right)\cdot$$

Taking for instance $\beta = \frac{1}{5}\log\alpha$ concludes the proof of Theorem 1.

Acknowledgements. We thank Persi Diaconis and Ohad Feldheim for helpful discussions and references. We thank Ryokichi Tanaka and Alex Lin Zhai for helpful comments and corrections. We also thank Ron Rivest for suggesting this problem to us.

References

1 Daniel Andrén, Lars Hellström, and Klas Markström. On the complexity of matrix reduction over finite fields. *Advances in applied mathematics*, 39(4):428–452, 2007.

2 Demetres Christofides. The asymptotic complexity of matrix reduction over finite fields. *arXiv preprint arXiv:1406.5826*, 2014.

3 Fan R. K. Chung and Ronald L. Graham. Stratified random walks on the n-cube. *Random Structures and Algorithms*, 11(3):199–222, 1997.

4 Persi Diaconis and Laurent Saloff-Coste. Walks on generating sets of abelian groups. *Probability theory and related fields*, 105(3):393–421, 1996.

5 Martin Kassabov. Kazhdan constants for $\mathrm{SL}_n(\mathbb{Z})$. *International Journal of Algebra and Computation*, 15(05n06):971–995, 2005.

6 D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov chains and mixing times.* AMS, 2009.

7 Aikaterini Sotiraki. *Authentication protocol using trapdoored matrices.* PhD thesis, Massachusetts Institute of Technology, 2016.

Lower Bounds for 2-Query LCCs over Large Alphabet*

Arnab Bhattacharyya[1], Sivakanth Gopi[2], and Avishay Tal[3]

1 Department of Computer Science and Automation, Indian Institute of Science Bangalore, Bangalore, India
arnabb@csa.iisc.ernet.in

2 Department of Computer Science, Princeton University, Princeton, NJ, USA
sgopi@cs.princeton.edu

3 School of Mathematics, Institute for Advanced Study, Princeton, NJ, USA
avishay.tal@gmail.com

Abstract

A locally correctable code (LCC) is an error correcting code that allows correction of any arbitrary coordinate of a corrupted codeword by querying only a few coordinates. We show that any 2-query locally correctable code $\mathcal{C} : \{0,1\}^k \to \Sigma^n$ that can correct a constant fraction of corrupted symbols must have $n \geqslant \exp(k/\log|\Sigma|)$ under the assumption that the LCC is *zero-error*. We say that an LCC is zero-error if there exists a non-adaptive corrector algorithm that succeeds with probability 1 when the input is an uncorrupted codeword. All known constructions of LCCs are zero-error.

Our result is tight upto constant factors in the exponent. The only previous lower bound on the length of 2-query LCCs over large alphabet was $\Omega((k/\log|\Sigma|)^2)$ due to Katz and Trevisan (STOC 2000). Our bound implies that zero-error LCCs cannot yield 2-server private information retrieval (PIR) schemes with sub-polynomial communication. Since there exists a 2-server PIR scheme with sub-polynomial communication (STOC 2015) based on a zero-error 2-query locally decodable code (LDC), we also obtain a separation between LDCs and LCCs over large alphabet.

1998 ACM Subject Classification E.4 Coding and Information Theory

Keywords and phrases Locally correctable code, Private information retrieval, Szemerédi regularity lemma

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.30

1 Introduction

In this work, we study error-correcting codes that are equipped with local algorithms. A code is called a locally correctable code (LCC) if there is a randomized algorithm which, given an index i and a received word w close to a codeword c in Hamming distance, outputs c_i by querying only a few positions of w. The maximum number of positions of w queried by the local correction algorithm is called the query complexity of the LCC.

The main problem studied regarding LCCs is the tradeoff between their query complexity and length. Intuitively, these two parameters enforce contrasting properties. Small query

[*] AB was partially supported by a DST Ramanujan Fellowship. SG was supported by NSF grants CCF-1523816, CCF-1217416 and part of this research was done while the author was at Microsoft Research, Redmond. AT was supported by the Simons Collaboration on Algorithms and Geometry, and by the National Science Foundation grant No. CCF-1412958.

complexity means that individual codeword symbols carry substantial information, while short length along with resilience to corruption means that information is spread out among the codeword symbols. In this paper, we explore one end of the spectrum of tradeoffs by studying 2-query locally correctable codes.

Also called "self-correction", the idea of local correction originated in works by Lipton [22] and by Blum and Kannan [7] on program checkers. In particular, [22, 3] used the fact that the Reed-Muller code is locally correctable to show average-case hardness of the Permanent problem. LCCs are closely related to locally decodable codes (LDCs), where the goal is to recover a symbol of the underlying message when given a corrupted codeword using a small number of queries [18]. LDCs are weaker than LCCs, in the sense that any LCC can be converted into an LDC while preserving relevant parameters (see Appendix A for a formal statement and proof). LDCs and LCCs have found applications in derandomization and hardness results [25, 15, 19]. See [29] for a detailed survey on LDCs and LCCs, as of 2010. In more recent years, the analysis of LDCs and LCCs has led to a greater understanding of basic problems in incidence geometry, the construction of design matrices and the theory of matrix scaling, e.g. [2, 14, 13].

One particularly important feature of LDCs is their tight connection to *information-theoretic private information retrieval (PIR)* schemes. PIR is motivated by the scenario where a user wants to retrieve an item from a database without revealing to the database owner what item he is asking for. Formally, the user wants to retrieve x_i from a k-bit database $\mathbf{x} = (x_1, \ldots, x_k)$. A trivial solution is for the database owner to transmit the entire database no matter what query the user has in mind, but this has a huge communication overhead. Chor et al. [8] observed that while with one database, nothing better than the trivial solution is possible, there are non-trivial PIR schemes if multiple servers can hold replicas of the database. It turns out that t-server PIR schemes with low communication are roughly equivalent to short t-query LDCs. More precisely, a 2-server PIR scheme for k bits of data with s bits of communication translates to a 2-query LDC $\mathcal{C} : \{0,1\}^k \to \Sigma^{2^s}$ where $\Sigma = \{0,1\}^s$. Note that in this translation, $|\Sigma|$ equals the length of the code.

Let $\mathcal{C} : \{0,1\}^k \to \Sigma^n$ be a 2-query LDC/LCC such that the corrector algorithm can tolerate corruptions at δn positions. Katz and Trevisan in their seminal work [18] showed that for 2-query LDCs, $n \geqslant \Omega(\delta(k/\log|\Sigma|)^2)$. (Since LDCs are weaker than LCCs, a lower bound on the length of LDCs also implies a lower bound on the length of LCCs). More than 15 years later, the Katz-Trevisan bound is still the best known for large alphabet Σ. However for small alphabet size, the dependence on k is shown to be exponential. Goldreich et al. [16] showed that $n \geqslant \exp(\delta k/|\Sigma|)$ for linear 2-query LDCs, while Kerenedis and de Wolf [20] (with further improvements in [28]) showed using quantum techniques that $n \geqslant \exp(\delta k/|\Sigma|^2)$ for arbitrary 2-query LDCs. But these lower bounds become trivial when $|\Sigma| = \Omega(n)$. However, the case of large alphabet $|\Sigma| \approx n$ is quite important to understand as this is the regime through which we would be able to prove lower bounds on the communication complexity of PIR schemes.

Given the lack of progress on LDC and PIR lower bounds, it is a natural question to ask whether strong lower bounds are possible for LCCs. In this work, we demonstrate an exponential improvement on the Katz-Trevisan bound for *zero-error LCCs.* We define a zero-error LCC to be an LCC for which the corrector algorithm is non-adaptive and succeeds with probability 1 when the input is an uncorrupted codeword. All current LCC constructions are zero-error, and in fact, any linear LCC can be made zero-error. We state our main theorem below informally, see Theorem 5 for a formal statement.

▶ **Theorem 1** (Informal). *If $\mathcal{C} : \{0,1\}^k \to \Sigma^n$ is a zero-error 2-query LCC that can correct δn corruptions, then $n \geqslant \exp(\mathrm{poly}(\delta) \cdot k/\log|\Sigma|)$.*[1]

1.1 Discussion of Main Result

The lower bound in Theorem 1 is tight in its dependence on k and Σ. Specifically, Yekhanin in the appendix of [4] gives the following elegant construction of a 2-query LCC $\mathcal{C} : \{0,1\}^k \to \Sigma^n$ with $n = 2^{O(k/\log|\Sigma|)}$ for any $\delta \leqslant 1/6, \Sigma$ and k. Assume $|\Sigma| = 2^b$ and $b \mid k$ for simplicity. Write $\mathbf{x} \in \{0,1\}^k$ as $(x_{i,j})_{i\in[b],j\in[k/b]}$. Then, for any $a \in [2^{k/b}]$, let $(\mathcal{C}(\mathbf{x}))_a = (\mathcal{H}(x_{i,1},\ldots,x_{i,k/b})_a : i \in [b]) \in \{0,1\}^b$ where $\mathcal{H}$ is the classical Hadamard encoding $\mathcal{H} : \{0,1\}^r \to \{0,1\}^{2^r}$ defined as $\mathcal{H}(\mathbf{y}) = (\sum_{i=1}^r y_i\xi_i \pmod 2) : \xi_1,\ldots,\xi_r \in \{0,1\})$. It is well-known that $\mathcal{H}$ is a 2-query LCC, and from this, it is easy to check that $\mathcal{C}$ is also. The parameters follow directly from the construction. A simple modification of this construction gives $(2^{O(\delta k/\log|\Sigma|)}/\delta)$-length 2-query LCCs that tolerate δn corruptions. The proof of Theorem 1 shows $n \geqslant \exp(\delta^4 k/\log|\Sigma|)$ which is therefore tight upto poly(δ) factors in the exponent.

The 2-query LCC described above is a linear code over $\mathbb{F}_{2^b}$. For linear codes $\mathcal{C} \subseteq \mathbb{F}_q^n$ (i.e., $\mathcal{C}$ is a linear subspace of $\mathbb{F}_q^n$), where $q = p^r$ for a prime p, [4] showed that $n \geqslant \exp(\delta k/r) = \exp(\delta k/\log_p|\Sigma|)$ where $k = \log|\mathcal{C}|$ is the message length and $|\Sigma| = p^r$. Thus, in terms of dependence on k and $|\Sigma|$, we extend the result of [4] from linear codes to all zero-error LCCs. Moreover, this work is much more elementary and simple than [4] which uses non-trivial results from additive combinatorics.

It is important to note that Theorem 1 cannot be true for 2-query LDCs. Such a result would contradict the construction in [12] of a zero-error 2-query LDC with $\log n = \log|\Sigma| = \exp(\sqrt{\log k}) = k^{o(1)}$ and $\delta = \Omega(1)$. So, our result can be interpreted as giving a separation between zero-error LCCs and LDCs over large alphabet. We conjecture that the zero-error restriction in the theorem can be removed, which if true, would yield the first separation between general LCCs and LDCs. It is still quite unclear what the correct lower bound for 2-query LDCs should look like. As mentioned above, Katz and Trevisan [18] show that $n \geqslant \Omega(\delta k^2/\log^2|\Sigma|)$. And the quantum arguments of [20, 28] give the lower bound $n \geqslant \exp(\delta k/|\Sigma|^2)$ which becomes trivial when $|\Sigma| = \Omega(n)$.

1.2 Proof Overview

Like most prior work on 2-query LDCs and LCCs, we view the query distribution of the local correcting algorithm as a graph. However, these previous works did not exploit the structure of the graph much beyond its size and degree, whereas our bound is due to a detailed use of the graph structure.

Let $\mathcal{C} : \{0,1\}^k \to \Sigma^n$ be a 2-query LCC. So, for every $i \in [n]$, there is a corrector algorithm $\mathcal{A}_i$ that when given access to $z \in \Sigma^n$ with Hamming distance at most δn from some codeword y, returns y_i with probability at least $2/3$. Assuming non-adaptivity, the algorithm $\mathcal{A}_i$ chooses its queries from a distribution on $[n]^2$. Katz and Trevisan [18] show how to extract a matching M_i of $\Omega(\delta n)$ disjoint edges on n vertices such that for any edge $e = (j,k)$ in M_i,

$$\mathop{\mathbf{Pr}}_{y}\left[\mathcal{A}_i(y) = y_i \mid \mathcal{A} \text{ queries } y \text{ at positions } j \text{ and } k\right] > \frac{1}{2} + \varepsilon$$

[1] An earlier version [5] of this paper showed that $n \geqslant \exp(c_\delta \cdot k/\log|\Sigma|)$ where c_δ has tower type dependence on δ due to the use of the Szemerédi regularity lemma.

for some constant $\varepsilon > 0$, where the probability is over a uniformly random codeword $y \in \mathcal{C}$. For zero-error LCCs, the situation is simpler in that essentially, for *every* codeword y and edge $e \in M_i$, $\mathcal{A}_i(y)$ returns y_i when it queries the elements of e. This is not exactly correct but let us suppose it's true for the rest of this section.

Let G be the union of $M_1, \ldots, M_n$. So, for every edge (j, k) in G, there is an i such that $(j, k) \in M_i$. Suppose our goal is to guess an unknown codeword c given the values of a small subset of coordinates of c. We assign labels in Σ to vertices of G corresponding to the subset of coordinates of c that we know already. Now, imagine a propagation process where we deduce the labels of unlabeled vertices by using the corrector algorithms. For example, if $(j, k) \in M_i$, j and k are labeled but i is not, we can use $\mathcal{A}_i$ to deduce the label at vertex i. Similarly, if $(x, y) \in M_u$ and $(u, v) \in M_w$, and x, y, v are labeled but u and w are not, we can run $\mathcal{A}_u$ to deduce the label of u and then $\mathcal{A}_w$ to deduce the label of w. The set of labels we infer will be the values of c at the corresponding coordinates. The goal of our analysis is to show that there is a set S of $O_\delta(\log n)^2$ vertices such that if the labels of S are known, then the propagation process can determine the labels of all n vertices. This immediately implies that the total number of codewords, 2^k, is at most $|\Sigma|^{|S|}$ and therefore, $k = O_\delta(\log n \cdot \log |\Sigma|)$. Instead, Katz and Trevisan [18] show that if you know the labels of $\sqrt{n}$ uniformly random coordinates, then you can recover the labels of most of the coordinates which leads to the bound $k = O_\delta(\sqrt{n} \cdot \log |\Sigma|)$. Intuitively, their lower bound is just one step of the propagation process.

The propagation process is perhaps more naturally described on a (directed) 3-uniform hypergraph where there is an edge (i, j, k) if $(j, k) \in M_i$. It "captures" i if (i, j, k) is an edge and j, k are already captured. Coja-Oghlan et al. [9] study exactly this process on random undirected 3-uniform hypergraphs in the context of constraint satisfaction problem solvers. Unfortunately, their techniques are specialized to random hypergraphs. The propagation process is also related to hypergraph peeling [23, 24], but again, most theoretical work is limited to random hypergraphs.

To motivate our approach, suppose $M_1, \ldots, M_n$ are each a perfect matching. For a set $S \subseteq [n]$, let $R(S)$ denote the set of vertices to which we can propagate starting from S. If $R(S) = [n]$, we are done. Otherwise, we show that we can double $|R(S)|$ by adding one more vertex to S. Note that for any $i \notin R(S)$, no edge in M_i can lie entirely inside $R(S)$, for then, i would also have been reached. So, each vertex in $R(S)$ must be incident to one edge in M_i for every $i \notin R(S)$. This makes the total number of edges between $R(S)$ and $[n] \setminus R(S)$ belonging to M_i for some $i \notin R(S)$ equal to $|R(S)| \cdot (n - |R(S)|)$. By averaging, there must be $j \notin R(S)$ that is incident to at least $|R(S)|$ edges, each belonging to some M_i for $i \notin R(S)$. Moreover, all these $|R(S)|$ edges must belong to matchings of different vertices. Hence, adding j to S doubles the size of $R(S)$. Hence, for some S of size $O(\log n)$, $R(S) = [n]$.

In the above special case (where all the matchings were perfect), we used the fact that the size of the cut between $R(S)$ and the rest of the graph is large and that many of these edges belong to M_i for $i \notin R(S)$. We observe that for any graph obtained from an LCC as above, this situation exists whenever $R(S)$ is not too large already and the minimum degree of every vertex in the graph is large (say, $\text{poly}(\delta) \cdot n$). This is because each vertex in $R(S)$ will be incident to many edges in matchings M_i for $i \notin R(S)$ (using the minimum degree requirement and that $|R(S)|$ is small) and such edges cannot have both endpoints inside $R(S)$ (as then $i \in R(S)$). So, indeed, there will be many edges with labels not in $R(S)$

[2] $O_\delta(\cdot)$ means that the involved constant can depend on δ.

crossing the cut, and averaging will yield a vertex whose addition to S will make $R(S)$ grow by a multiplicative factor. Therefore, if the minimum degree requirement is met, we can keep repeating this process until $R(S)$ becomes large, of size $\mathrm{poly}(\delta) \cdot n$. Now, in a key lemma of our proof, we show that for any graph obtained from an LCC as above, we can greedily find a subset of the vertices V' such that the the subgraph induced by the vertices of V' and the edges labeled by V' has large minimum degree. So, we can repeatedly apply the above argument to V' to find a subset S of size $O_\delta(\log n)$ such that $R(S)$ contains $\mathrm{poly}(\delta) \cdot n$ vertices.

Recall that our goal is to find a small set S such that $R(S) = [n]$. So, at this stage, we would ideally like to continue the argument on $V'' = [n] \setminus R(S)$. The only issue we can face is that the graph on V'' restricted to edges labeled by V'' may not have the LCC structure. Indeed, it could be that most edges labeled by V'' are not spanned by vertices in V''. However in this case, there will be a vertex u in V'' incident to many V''-labeled edges that have their other endpoints in $R(S)$, so that we can increase $R(S)$ by adding u to S. Thus, either $R(S)$ may be grown directly or else the rest of the vertices looks approximately like an LCC, so that we can recurse. Modulo some important technical details, our proof is now complete[3].

The zero-error assumption seems necessary to make the propagation process well-defined. Otherwise, for each labeled vertex, there is some probability that the label is incorrect for the codeword in question. But since there may be $\Omega(\log n) = \omega(1)$ steps of propagation, the error probability may blow up by this factor. So, it seems we need different techniques to handle correctors that have constant probability of error when the input is a codeword. One possibility is using information theory to better handle the spread of error[4].

2 Zero-error 2-query LCCs

We begin by formally defining zero-error 2-query LCCs.

▶ **Definition 2.** Let Σ be some finite alphabet and let $\mathcal{C} \subset \Sigma^n$ be a set of codewords. $\mathcal{C}$ is called a $(2, \tau)$-LCC with zero-error if there exists a randomized algorithm $\mathcal{A}$ such that following is true:

1. $\mathcal{A}$ is given oracle access to some $z \in \Sigma^n$ and an input $i \in [n]$. It outputs a symbol in Σ after making at most 2 non-adaptive queries to z.
2. If $z \in \Sigma^n$ is τ-close to some codeword $c \in \mathcal{C}$ in Hamming distance, then for every $i \in [n]$, $\mathbf{Pr}[\mathcal{A}^z(i) = c_i] \geqslant 2/3$.
3. If $c \in \mathcal{C}$, then for every $i \in [n]$, $\mathbf{Pr}[\mathcal{A}^c(i) = c_i] = 1$ i.e. if the received word has no errors, then the local correction algorithm will not make any error.

Note that the above definition differs from the standard notion of non-adaptive 2-query LCCs only in part (3) above. The choice of 2/3 in part (2) of the definition above is somewhat arbitrary. We can make it any constant greater than 1/2. More generally, it is only required

[3] An earlier version [5] of this paper had a different argument for the main theorem, based on a "decomposition theorem" proved using the Szeméredi regularity lemma for directed graphs [26, 1]. The idea was to partition the graph into a constant number of edge expanders. In each such part, the sizes of cuts are large and so the propagation process can be easily analyzed. The proof given here is simpler and yields much better dependence on δ. However, because the decomposition theorem for directed graphs may be of general interest, we have included it in Appendix B of this paper.

[4] This approach is taken in [17] to prove an exponential lower bound for smooth 2-query LDCs over binary alphabet when the decoder has subconstant error probability. Jain's analysis seems to work only for binary codes but is similar in spirit to ours.

that for every $\sigma \neq c_i, \mathbf{Pr}[\mathcal{A}^z(i) = c_i] > \mathbf{Pr}[\mathcal{A}^z(i) = \sigma] + \varepsilon$ for some $\varepsilon > 0$, i.e., c_i should win the plurality vote among all symbols by a constant margin.

We next show that the corrector for any zero-error LCC can be brought into a "normal" form. A similar statement is known for general LDCs and LCCs [18, 29] but we need to be a bit more careful because we want to preserve the zero-error property. Note that the proof overview in Section 1.2 assumed that the set T_1 below is empty.

▶ **Lemma 3.** *Let $\mathcal{C} \subset \Sigma^n$ be a $(2, \tau)$-LCC with zero error. Then, there exists a partition of $[n] = T_1 \cup T_2$ such that:*

1. *For every $i \in T_1$, there exists a distribution $\mathcal{D}_i$ over $[n] \cup \{\phi\}$ and algorithms $\mathcal{R}^i_j$ for every $j \in [n] \cup \{\phi\}$ such that for every codeword $c \in \mathcal{C}$,*

$$\mathop{\mathbf{Pr}}_{j \sim \mathcal{D}_i} \left[\mathcal{R}^i_j(c_j) = c_i\right] \geqslant \frac{2}{3}.^5$$

Moreover the distribution $\mathcal{D}_i$ is smooth over $[n]$ i.e. for every $j \in [n]$, $\mathbf{Pr}_{\mathcal{D}_i}[j] \leqslant \frac{4}{\tau n}$.

2. *For every $i \in T_2$, there exists a matching $\mathcal{M}_i$ of edges in $[n] \setminus \{i\}$ of size $|\mathcal{M}_i| \geqslant \frac{\tau}{4} n$ such that: For every $c \in \mathcal{C}$, c_i can be recovered from (c_j, c_k) for any $(j,k) \in \mathcal{M}_i$ i.e. there exists algorithms $\mathcal{R}^i_{j,k}$ for every edge $(j,k) \in \mathcal{M}_i$ such that for every $c \in \mathcal{C}$,*

$$\mathcal{R}^i_{j,k}(c_j, c_k) = c_i.$$

Proof. Fix $\varepsilon = \tau/4$. Let $\mathcal{A}$ be the local corrector algorithm for $\mathcal{C}$ and let $\mathcal{Q}_i$ be the distribution over 2-tuples of $[n]$ corresponding to the queries $\mathcal{A}(i)$ makes to correct coordinate i.[6] Let $\mathrm{supp}(\mathcal{Q}_i)$ be the set of edges in the support of $\mathcal{Q}_i$. We have two cases:

Case 1: $\mathrm{supp}(\mathcal{Q}_i)$ contains a matching of size εn.

In this case, we include $i \in T_2$ and define $\mathcal{M}_i$ to be a matching of size εn in $\mathrm{supp}(\mathcal{Q}_i)$. Let $\mathcal{R}^i_{j,k}(z_j, z_k)$ be the output[7] of $\mathcal{A}^z(i)$ when it samples (j,k) from the distribution $\mathcal{Q}_i$. So we have for every $\sigma \in \Sigma$,

$$\mathop{\mathbf{Pr}}_{(j,k) \sim \mathcal{Q}_i} [\mathcal{R}^i_{j,k}(z_j, z_k) = \sigma] = \mathbf{Pr}[\mathcal{A}^z(i) = \sigma].$$

Now since our LCC is zero-error, for every $(j,k) \in \mathrm{supp}(\mathcal{Q}_i)$, we have $\mathcal{R}^i_{j,k}(c_j, c_k) = c_i$. This takes care of part (2).

Case 2: $\mathrm{supp}(\mathcal{Q}_i)$ doesn't contain a matching of size εn.

In this case we include $i \in T_1$. Since $\mathrm{supp}(\mathcal{Q}_i)$ doesn't contain a matching of size εn, there exists a vertex cover of size at most $2\varepsilon n$, say V_i. Also define $B_i \subset [n]$ to be the set of vertices which are queried with high probability by $\mathcal{A}^z(i)$ i.e.

$$B_i = \left\{ j : \mathbf{Pr}[\mathcal{A}^z(i) \text{ queries } j] \geqslant \frac{1}{\varepsilon n} \right\}.$$

Clearly $|B_i| \leqslant 2\varepsilon n$ because $\mathcal{A}^z(i)$ makes at most two queries. We now define a new one-query corrector for i, $\tilde{\mathcal{A}}^z(i)$ as follows: simulate $\mathcal{A}^z(i)$, but whenever $\mathcal{A}^z(i)$ queries z at a coordinate in $V_i \cup B_i$, $\tilde{\mathcal{A}}^z(i)$ doesn't query that coordinate and assumes that the queried coordinate is 0 (or some fixed symbol in Σ). Note that $\tilde{\mathcal{A}}^z(i)$ makes at most one query to z since V_i is a vertex cover for the support of $\mathcal{Q}_i$. Also $\tilde{\mathcal{A}}^c(i)$ behaves exactly

[6] Wlog, we can assume $\mathcal{A}(i)$ always queries two coordinates.

[7] Note that $\mathcal{R}^i_{j,k}$ might use additional randomness.

like $\mathcal{A}^{c'}(i)$ where c' is the word formed by zeroing out the $V_i \cup B_i$ coordinates of c. Since $|V_i \cup B_i| \leqslant 4\varepsilon n \leqslant \tau n$, we have

$$\mathbf{Pr}[\tilde{\mathcal{A}}^c(i) = c_i] = \mathbf{Pr}[\mathcal{A}^{c'}(i) = c_i] \geqslant \frac{2}{3}.$$

Now define the distribution $\mathcal{D}_i$ over $[n] \cup \{\phi\}$ as:

$$\mathop{\mathbf{Pr}}_{\mathcal{D}_i}[j] = \mathbf{Pr}[\tilde{\mathcal{A}}^z(i) \text{ queries } j]$$

for $j \in [n]$ and

$$\mathop{\mathbf{Pr}}_{\mathcal{D}_i}[\phi] = \mathbf{Pr}[\tilde{\mathcal{A}}^z(i) \text{ doesn't make any query}].$$

Since we never query elements of B_i, we have the required smoothness i.e. $\mathbf{Pr}_{\mathcal{D}_i}[j] \leqslant 1/(\varepsilon n)$ for all $j \in [n]$. Also define $\mathcal{R}^i_j(z_j)$ to be the output (can be randomized) of $\tilde{\mathcal{A}}^z(i)$ when it queries $j \in [n]$ and $\mathcal{R}^i_\phi(c_\phi)$ to be the output (can be randomized) of $\tilde{\mathcal{A}}^z(i)$ when it doesn't make any query where c_ϕ is an empty input defined for ease of notation. By definition, we have

$$\mathop{\mathbf{Pr}}_{j \sim \mathcal{D}_i}[\mathcal{R}^i_j(c_j) = c_i] = \mathbf{Pr}[\tilde{\mathcal{A}}^c(i) = c_i] \geqslant \frac{2}{3}.$$

This proves part (1). ◀

3 Proof of lower bound

3.1 An information theoretic lemma

The proof of Theorem 1 works by showing that there is randomized algorithm which can guess an unknown codeword $c \in \mathcal{C} \subset \Sigma^n$ with high probability by making a small number of queries. From this we would like to show that $|\mathcal{C}|$ cannot be large. We will apply Fano's inequality which is a basic information theoretic inequality to achieve this. We will assume familiarity with basic notions in information theory; we refer the reader to [10] for precise definitions and the proofs of the facts we use. Given random variables X, Y, Z, let $H(X)$ be the entropy of X which is the amount of information contained in X. $H(X|Y)$ is the conditional entropy of X given Y which is the amount of information left in X if we know Y. The mutual information $I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$ is the amount of common information between X, Y. If X, Y are independent, then $I(X;Y) = 0$. The conditional mutual information $I(X;Y|Z)$ is the mutual information between X, Y given Z. We have the following chain rule for mutual information:

$$I(X;YZ) = I(X;Z) + I(X;Y|Z).$$

We also need the following basic inequality:

$$I(X;Y|Z) \leqslant H(X|Z) \leqslant \log|\mathcal{X}|$$

where $\mathcal{X}$ is the support of the random variable X. We will now state Fano's inequality which says that if we can predict X very well from Y i.e. there is a predictor $\hat{X}(Y)$ such that $\mathbf{Pr}[\hat{X}(Y) \neq X] \leqslant p_e$ where p_e is small, then $H(X|Y)$ should be small as well (see [10] for a proof). More precisely,

$$H(X|Y) \leqslant h(p_e) + p_e \log(|\mathcal{X}| - 1) \qquad \text{(Fano's inequality)}$$

where $h(x) = -x\log x - (1-x)\log(1-x)$ is the binary entropy function and $\mathcal{X}$ is the support of random variable X.

▶ **Lemma 4.** *Suppose there exists a randomized algorithm $\mathcal{P}$ such that for every $c \in \mathcal{C} \subset \Sigma^n$, given oracle access to c, $\mathcal{P}$ makes at most t queries to c and outputs c with probability $\geqslant 1/2$, then* $\log|\mathcal{C}| \leqslant O(t \log|\Sigma|)$.

Proof. Let X be a random variable which is uniformly distributed over $\mathcal{C}$. Let R be the random variable corresponding to the random string of the algorithm $\mathcal{P}$ and let $S(R)$ be the set of coordinates queried by $\mathcal{P}$ when the random string is R. We can guess the value of X with probability $\geqslant 1/2$ given $X_{S(R)}, R$ where $X_{S(R)}$ is the restriction of X to $S(R)$. By Fano's inequality,

$$H(X \mid X_{S(R)}, R) \leqslant h(1/2) + \frac{1}{2} \cdot \log(|\mathcal{C}| - 1) \leqslant 1 + \frac{1}{2} \log|\mathcal{C}|.$$

We can bound the mutual information between X and $X_{S(R),R}$ as follows:

$$\begin{aligned} I(X; X_{S(R)}, R) &= I(X;R) + I(X; X_{S(R)}|R) && \text{(Chain rule for mutual information.)} \\ &\leqslant 0 + H(X_{S(R)}|R) && \text{(Since } X \text{ and } R \text{ are independent.)} \\ &\leqslant t \log|\Sigma|. \end{aligned}$$

But we also have

$$I(X; X_{S(R)}, R) = H(X) - H(X|X_{S(R)}, R) \geqslant \log|\mathcal{C}| - \frac{1}{2}\log|\mathcal{C}| - 1 \geqslant \frac{1}{2}\log|\mathcal{C}| - 1.$$

Combining the upper and lower bound for $I(X; X_{S(R)}, R)$, we get the required bound. ◀

3.2 Proof of Theorem 1

The following is a restatement of Theorem 1.

▶ **Theorem 5.** *Let $\mathcal{C} \subset \Sigma^n$ be a $(2,\tau)$-LCC which is zero-error, then*

$$|\mathcal{C}| \leqslant \exp\left(O(\tfrac{1}{\tau^4} \cdot \log n \cdot \log|\Sigma|)\right).$$

Proof. We will construct a randomized algorithm $\mathcal{P}$ such that for every $c \in \mathcal{C}$, given oracle access to c, $\mathcal{P}$ makes at most $O(\frac{1}{\tau^4} \cdot \log n)$ queries to c and outputs c with probability $\geqslant 1 - 1/n$. By Lemma 4, we get the required bound.

Let $[n] = T_1 \cup T_2$ be partition of coordinates given by Lemma 3.

▶ **Claim 6.** *Algorithm $\mathcal{P}$ can learn $c|_{T_1}$ with probability $\geqslant 1 - 1/n$ by querying a uniformly random (sampled with repetitions) subset S of size $r = O(\frac{1}{\tau^2} \cdot \log n)$.*

Proof. Let $S = \{Z_1, \cdots, Z_r\}$ where each Z_i is a uniformly random element of $[n]$. By Lemma 3, for every $u \in T_1$, we have a smooth distribution $\mathcal{D}_u$ over $[n]$ and algorithms $\mathcal{R}_v^u$ for every $v \in [n]$. Let's fix $u \in T_1$ and let $p_v = \mathbf{Pr}_{\mathcal{D}_u}[v]$. By smoothness, $p_v \leqslant \frac{4}{\tau n}$ for every $v \in [n]$. The algorithm $\mathcal{P}$ estimates c_u as follows: Define the weight of σ to be

$$W_\sigma = p_\phi \cdot \mathbf{Pr}[\mathcal{R}_\phi^u = \sigma] + \frac{1}{r}\sum_{i=1}^{r} np_{Z_i} \cdot \mathbf{Pr}[\mathcal{R}_{Z_i}^u(c_{Z_i}) = \sigma]$$

and output the symbol with the maximum weight. We will show that

$$\mathbf{Pr}[\mathcal{P} \text{ guesses } c_u \text{ incorrectly}] \leqslant \frac{1}{n^2}.$$

For $\sigma \in \Sigma$ and $v \in [n] \cup \{\phi\}$, let $f_v^\sigma = \mathbf{Pr}[\mathcal{R}_v^u(c_v) = \sigma]$. The weight of σ is given by

$$W_\sigma = p_\phi f_\phi^\sigma + \frac{1}{r}\sum_{i=1}^{r} np_{Z_i} f_{Z_i}^\sigma.$$

We can calculate the expected value of the weight as

$$\begin{aligned}\mathbf{E}[W_\sigma] &= p_\phi f_\phi^\sigma + \mathbf{E}[np_{Z_1} f_{Z_1}^\sigma]\\ &= p_\phi \,\mathbf{Pr}[\mathcal{R}_\phi^u(c_\phi) = \sigma] + \sum_{v\in[n]} p_v \,\mathbf{Pr}[\mathcal{R}_v^u(c_v) = \sigma] = \mathop{\mathbf{Pr}}_{v\sim\mathcal{D}_u}[\mathcal{R}_v^u(c_v) = \sigma].\end{aligned}$$

Therefore W_σ is an unbiased estimator for $\mathbf{Pr}_{v\sim\mathcal{D}_u}[\mathcal{R}_v^u(c_v) = \sigma]$. Also $p_{Z_i} \leqslant \frac{4}{\tau n}$ and $f_{Z_i}^\sigma \leqslant 1$, so $np_{Z_i} f_{Z_i}^\sigma \leqslant \frac{4}{\tau}$. Applying Hoeffding's inequality,

$$\mathbf{Pr}\left[|W_\sigma - \mathbf{E}[W_\sigma]| \geqslant \frac{1}{20}\right] \leqslant \exp\left(-\Omega(r\tau^2)\right) \leqslant 1/2n^2$$

when $r \gg \frac{1}{\tau^2}\log n$. By Lemma 3,

$$\mathbf{E}[W_{c_u}] = \mathop{\mathbf{Pr}}_{v\sim\mathcal{D}_u}[\mathcal{R}_v^u(c_v) = c_u] \geqslant \frac{2}{3}.$$

Therefore, $\mathbf{Pr}[W_{c_u} \leqslant \frac{2}{3} - \frac{1}{20}] \leqslant 1/2n^2$. Now we will show that no other symbol can have higher weight than W_{c_u} except with probability $\frac{1}{2n^2}$. For this let us look at

$$\begin{aligned}\sum_{\sigma\in\Sigma} W_\sigma &= \sum_\sigma p_\phi f_\phi^\sigma + \frac{1}{r}\sum_{i=1}^{r} np_{Z_i}\sum_\sigma f_{Z_i}^\sigma\\ &= p_\phi \sum_\sigma \mathbf{Pr}[\mathcal{R}_\phi^u = \sigma] + \frac{1}{r}\sum_{i=1}^{r} np_{Z_i}\sum_\sigma \mathbf{Pr}[\mathcal{R}_{Z_i}^u(c_{Z_i}) = \sigma]\\ &= p_\phi + \frac{1}{r}\sum_{i=1}^{r} np_{Z_i}\end{aligned}$$

So $\mathbf{E}[\sum_{\sigma\in\Sigma} W_\sigma] = p_\phi + \mathbf{E}[np_{Z_1}] = 1$ and $np_{Z_i} \leqslant \frac{4}{\tau}$. Therefore by Hoeffding's inequality applied again, we get

$$\mathbf{Pr}\left[\left|\sum_{\sigma\in\Sigma} W_\sigma - 1\right| \geqslant \frac{1}{20}\right] \leqslant \exp\left(-\Omega(r\tau^2)\right) \leqslant \frac{1}{2n^2}$$

when $r \gg \frac{1}{\tau^2}\log n$. So with probability $\geqslant 1 - \frac{1}{n^2}$, we have $W_{c_u} \geqslant \frac{2}{3} - \frac{1}{20}$ and $\sum_{\sigma\in\Sigma} W_\sigma \leqslant 1 + \frac{1}{20}$. Therefore with probability $\geqslant 1 - \frac{1}{n^2}$, c_u will be the symbol with maximum weight and the algorithm $\mathcal{P}$ will guess c_u correctly with probability $\geqslant 1 - \frac{1}{n^2}$. By union bound, we get that $\mathcal{P}$ can guess c_u correctly for all $u \in T_1$ with probability $\geqslant 1 - \frac{1}{n}$. ◀

We will now show that after learning $c|_{T_1}$, $\mathcal{P}$ can now learn $c|_{T_2}$ by querying a further $O_\tau(\log n)$ coordinates from c and this process will be deterministic i.e. no further randomness is needed. Define $R(S)$ to be the set of coordinates of c that can be recovered correctly given $c|_S$. In Claim 6, we have shown that if S is a randomly chosen subset of size $O_\tau(\log n)$, then $T_1 \subseteq R(S)$ with probability $\geqslant 1 - \frac{1}{n}$. From now on we assume that $\mathcal{P}$ has already recovered coordinates of T_1 correctly i.e. $T_1 \subseteq R(S)$. If $T_2 \subseteq R(S)$ then we are done, the algorithm $\mathcal{P}$ can output the entire c with probability $\geqslant 1 - \frac{1}{n}$. So we can assume that $T_2 \not\subseteq R(S)$. Our goal is to show that we can add a further $O(\text{poly}(1/\tau) \cdot \log n)$ vertices to S and have $R(S) = V = T_1 \cup T_2$. We show that this is indeed the case in the next section by proving the following claim, which completes the proof.

▶ **Claim 7.** *There exists a set S of size $O((1/\tau)^4 \cdot \log n)$ such that $R(S \cup T_1) = V$.* ◀

3.3 Proof of Claim 7

Claim 7 is purely graph theoretical. Let $G = (V, E)$ be the graph with $V = [n] = T_1 \cup T_2$ and $E = \cup_{i \in T_2} \mathcal{M}_i$ where $\mathcal{M}_i$ are partial matchings of size at least $(\tau/4)n$ given by Lemma 3. Let $\delta := \tau/4$. We will label each edge in E with a label in T_2 indicating which matching it belongs to. We can have parallel edges in E, but they will have different labels since they belong to different matchings. Recall that $R(S)$ is the set of coordinates of c that can be inferred from $c|_S$. Lemma 3 implies the following closure property for $R(S)$: if $(i, j) \in \mathcal{M}_k$ and $i, j \in R(S)$ then $k \in R(S)$. Next, we define $R(S)$ formally based on the graph G using this closure property.

▶ **Definition 8.** Let $G = (V, E)$ as above. Let $S \subseteq V$. We define the set $R_G(S) \subseteq V$ to be the smallest set of vertices such that:

1. $S \subseteq R_G(S)$
2. For all $i, j \in R_G(S)$ and $k \in [n]$, if $(i, j) \in \mathcal{M}_k$, then $k \in R_G(S)$. (In words, if there exists an edge (i, j) in the graph G labeled with k and both i and j are in $R_G(S)$, then so is k.)

(When the context is clear, we will use $R(S)$ instead of $R_G(S)$.) Our goal is to show that in any graph G as above, there exists a set $S \subseteq V$ of size $\text{poly}(1/\delta) \cdot \log(n)$ such that $R_G(S \cup T_1) = V$. As a first step, we get rid of the set T_1, by showing that proving the claim in the case $T_1 = \emptyset$ implies Claim 7 for any other set. To see that observe that if we take G' to be the union of G with a collection of partial matching $\{\mathcal{M}_j\}_{j \in T_1}$, then $R_{G'}(S) \subseteq R_G(S \cup T_1)$ for any set $S \subseteq V$. Thus, it suffices to introduce dummy matchings $\{\mathcal{M}_j\}_{j \in T_1}$ for each $\mathcal{M}_j$ of size δn, and prove that there exists a set S of size $\text{poly}(1/\delta) \cdot \log(n)$ such that $R_{G'}(S) = V$.

▶ **Claim 9** (Claim 7, case $T_1 = \emptyset$, restated). *Let $G = (V, E)$ be a graph with $V = [n]$ and $E = \mathcal{M}_1 \cup \cdots \cup \mathcal{M}_n$ where each $\mathcal{M}_i$ is a partial matching of size at least δn. Then, there exists a subset $S \subseteq V$ of size $O((1/\delta)^4 \cdot \log n)$ such that $R_G(S) = V$.*

From here henceforth we assume (without loss of generality) that $T_1 = \emptyset$ and $T_2 = [n]$, and prove Claim 9. The following lemma tells us that we can find a subgraph G' of G such that each vertex in G' has high degree. Note that the lemma finds a subgraph restricted to a set of vertices V', and also restricted to the set of edges labeled with V'.

We shall use this lemma inductively. During induction, we will remove some edges from the matchings. Thus, instead of asserting that all matchings are of size at least $\delta|V|$, we assume that all but $0.1\delta|V|$ of the matchings have at least $0.9\delta|V|$ edges.

▶ **Lemma 10** (Clean-Up Lemma). *Let $G = (V, E)$ be a graph with a finite set of vertices V and $E = \bigcup_{i \in V} \mathcal{M}_i$, where each $\mathcal{M}_i$ is a partial matching on V. Assume all but $0.1\delta|V|$ of the matchings $\mathcal{M}_i$ have size at least $0.9\delta|V|$. Then, there exists a subset $V' \subseteq V$ of size at least $\delta \cdot |V|$ so that the graph $G' = (V', E')$ where $E' = \bigcup_{i \in V'} \mathcal{M}_i \cap (V' \times V')$ has minimal degree at least $(\delta^2/4) \cdot |V|$.*

Proof. We find the set V' greedily. Let $\delta' := \delta^2/4$. Initialize $V' = V$. If the minimum degree in the remaining graph on V' is at least $\delta' \cdot |V|$ then we stop. Otherwise, remove the vertex $i \in V'$ with minimal degree, and remove all edges labeled i. We repeat this process until no vertices of degree smaller than $\delta' \cdot |V|$ exist.

If the process stopped when $|V'| \geqslant \delta|V|$ then we are done. We are left to show that the process cannot proceed past this point. Let's assume by contradiction that we can continue the process after this point. As we decrease the size of V' by one in each iteration, we must reach at a certain point of the process to a set of vertices $V' = V^*$ of size exactly $\delta|V|$.

Denote by

$$E^*(V') := \bigcup_{i\in V^*} \mathcal{M}_i \cap (V' \times V').$$

Next, we upper and lower bound $|E^*(V^*)|$ to derive a contradiction.

The upper bound $|E^*(V^*)| \leqslant |V^*|\cdot|V^*|/2$ follows since the edges $E^*(V^*)$ form a collection of $|V^*|$ partial matchings on V^*. To lower bound $|E^*(V^*)|$ we use the properties of the greedy process. The initial size of the set $E^*(V')$ (when $V' = V$) is at least $0.9\delta|V|\cdot(|V^*|-0.1\delta|V|) \geqslant 0.9^2\delta^2\cdot|V|^2$. In every iteration, we remove at most $\delta'|V|$ edges from this set of edges. As there are at most $|V|$ steps, we are left with at least $0.9^2\delta^2|V|^2 - \delta'|V|^2$ edges, i.e., $|E^*(V^*)| \geqslant 0.9^2\delta^2|V|^2 - \delta'|V|^2$. Combining both upper and lower bounds on $|E^*(V^*)|$ gives

$$\frac{1}{2}\cdot\delta^2\cdot|V|^2 \geqslant |E^*(V^*)| \geqslant (0.9^2\delta^2 - \delta')\cdot|V|^2 = (0.9^2\delta^2 - \delta^2/4)\cdot|V|^2$$

which yields a contradiction since $1/2 < 0.9^2 - 1/4$. ◀

▶ **Lemma 11** (Exponentially growing a set of known coordinates). *Let $G = (V, E)$ be a graph with V and $E = \bigcup_{i\in V}\mathcal{M}_i$ such that each $v \in V$ has degree at least d. Then, there exists a subset $S \subseteq V$ of size at most $O((|V|/d)\cdot\log|V|)$ with $|R(S)| \geqslant d/2$.*

Proof. We pick the set $S \subseteq V$ iteratively, picking one element in each step. We start with $S = \{v\}$ for some arbitrary $v \in V$.

Assume we picked t elements so far for the set S. If $|R(S)| \geqslant d/2$, then we are done. Otherwise, by the definition of $R(S)$, for any $i \in V \setminus R(S)$, none of the edges in the matching $\mathcal{M}_i$ is inside $R(S)$. We wish to show that there exists an $i \in V \setminus R(S)$ with many edges into $R(S)$ marked with labels outside $R(S)$. Then, we will add i to S, which will reveal a lot of new coordinates.

For two disjoint sets of vertices $A, B \subseteq V$ we denote by $E(A, B)$ the set of edges between A and B in the graph G. If A consists of one element, i.e., $A = \{a\}$ we denote $E(a, B) = E(A, B)$. Let $A = R(S)$. Let $B = V \setminus A$. We have

$$\left|E(A,B)\cap\bigcup_{i\in B}\mathcal{M}_i\right| = \sum_{a\in A}\left|E(a,B)\cap\bigcup_{i\in B}\mathcal{M}_i\right| = \sum_{a\in A}\left|E(a,V\setminus\{a\})\cap\bigcup_{i\in B}\mathcal{M}_i\right| \tag{1}$$

where the last equality follows since there are no edges labeled $i \in B$ between any two vertices in A. For each $a \in A$ there are at least d edges touching a and at most $|A|$ of them appeared in $\bigcup_{i\in A}\mathcal{M}_i$, hence $\left|E(a, V\setminus\{a\})\cap\bigcup_{i\in B}\mathcal{M}_i\right| \geqslant d - |A| \geqslant d/2$. Plugging this estimate to Eq. (1) gives

$$\left|E(A,B)\cap\bigcup_{i\in B}\mathcal{M}_i\right| \geqslant |A|\cdot d/2\ .$$

By averaging there exists a vertex $b \in B$ with at least $|A|\cdot\frac{d}{2|V|}$ edges to A labeled with B. So as long as $|A| = |R(S)| \leqslant d/2$ we are extending the set $R(S)$ by at least $|R(S)|\cdot\frac{d}{2|V|}$ elements, i.e. by a multiplicative factor of $(1+\frac{d}{2|V|})$. Hence, after t iterations, either $|R(S)| \geqslant (1+\frac{d}{2|V|})^t$ or $|R(S)| \geqslant d/2$. Taking $t = O(\frac{|V|}{d}\cdot\log|V|)$ gives that after at most t iterations $|R(S)| \geqslant d/2$. ◀

▶ **Lemma 12** (Covering $1-\delta$ fraction of the coordinates implies covering all coordinates). *Let $G = (V, E)$ be a graph with $V = [n]$ and $E = \mathcal{M}_1 \cup \mathcal{M}_2 \cup \ldots \cup \mathcal{M}_n$ and each $\mathcal{M}_i$ is a partial matching of size at least δn. Let $S \subseteq V$. If $|R(S)| > (1-\delta)n$, then $R(S) = V$.*

Proof. Let $v \in V$. We show that there is an edge inside $R(S)$ marked v. Indeed, there are at least δn edges labeled v and they form a partial matching. If $|V \setminus R(S)| < \delta n$, one of these edges do not touch $(V \setminus R(S))$, i.e., it is an edge connecting two vertices in $R(S)$. ◀

▶ **Lemma 13** (Two Cases). *Let $G = (V, E)$ be a graph with $V = [n]$ and $E = \mathcal{M}_1 \cup \mathcal{M}_2 \cup \ldots \cup \mathcal{M}_n$ where each $\mathcal{M}_i$ is a partial matching of size at least δn. Let $S \subseteq V$. Assume $|R(S)| \leqslant (1-\delta)n$. Then, either*

1. *There exists an $i \in V \setminus R(S)$ such that $|R(S \cup \{i\})| \geqslant |R(S)| + 0.01 \cdot \delta^2 \cdot n$.*
2. *In the graph $G' = (V', E')$ with $V' = V \setminus R(S)$ and $E' = \bigcup_{i \in V'} \mathcal{M}_i \cap (V' \times V')$ all but at most $0.1\delta \cdot |V'|$ of the matchings have at least $0.9\delta \cdot n$ edges.*

Proof. Recall that the labels of edges incident to any vertex i are distinct, since the graph is a union of partial matchings. Denote by $A = R(S)$ and $B = V \setminus R(S)$. Assume for any $i \in B$ there are at most $0.01\delta^2 \cdot n$ edges to A labeled with labels in B. (Otherwise, extend S by i and get $|R(S \cup \{i\})| \geqslant |R(S)| + 0.01\delta^2 \cdot n$.) Then, there are at most $0.01\delta^2 \cdot n \cdot |B|$ edges in the cut (A, B) with labels in B. By definition of $A = R(S)$, there are no edges between A and A labeled with B. Thus, at most $0.01\delta^2 n \cdot |B|$ edges are missing from the matchings labeled by B if we restrict to edges between B and B. Hence, at most $0.1\delta \cdot |B|$ of the matchings may miss more than $0.1\delta \cdot n$ of their edges. ◀

We are now ready to prove Claim 9.

Proof of Claim 9. Initialize $S := \emptyset$. We repeat the following process. While $R(S) \neq V$, check if there exists $i \in V \setminus R(S)$ such that $|R(S \cup \{i\})| \geqslant |R(S)| + 0.01\delta^2 n$. We have two cases:

1. If such an i exists, update $S := S \cup \{i\}$.
2. Else, let $G' = (V', E')$ where $V' = V \setminus R(S)$ and $E' = \bigcup_{i \in V'} \mathcal{M}_i \cap (V' \times V')$. Let $M'_i := \mathcal{M}_i \cap (V' \times V')$. By Lemma 12, $|V'| \geqslant \delta n$. By Lemma 13, all but at most $0.1\delta|V'|$ of the matchings M'_i for $i \in V'$ have at least $0.9\delta n$ edges. Denote by $\delta' = 0.9\delta n/|V'| \geqslant \delta$. We apply Lemma 10 on G' to get a subgraph $G'' = (V'', E'')$ defined by a subset V'' of size $\Omega(\delta'|V'|)$ and $E'' = \bigcup_{i \in V''} \mathcal{M}_i \cap (V'' \times V'')$ with minimal degree $d = \Omega((\delta')^2 \cdot |V'|) \geqslant \Omega(\delta^2 n)$. We apply Lemma 11 on G'' to get a set $S'' \subseteq V''$ of size $O(\log |V''| \cdot (|V''|/d)) = O(\log n \cdot (1/\delta')^2)$ with $|R_{G''}(S'')| \geqslant \Omega(d) \geqslant \Omega(\delta^2 n)$. We update $S := S \cup S''$.

The number of times we apply case 1 or case 2 is at most $O(1/\delta^2)$, since each such step introduces $\Omega(\delta^2 n)$ new vertices to $R(S)$. In each application of case 2, at most $O((1/\delta')^2 \cdot \log n) \leqslant O((1/\delta^2) \cdot \log n)$ elements are added to S. Overall, the size of S at the end of the process will be

$$O\left(\tfrac{1}{\delta^2}\right) + O\left(\tfrac{1}{\delta^2} \cdot \tfrac{1}{\delta^2} \cdot \log n\right) = O\left(\tfrac{1}{\delta^4} \cdot \log n\right) . \quad ◀$$

References

1 Noga Alon and Asaf Shapira. Testing subgraphs in directed graphs. *Journal of Computer and System Sciences*, 3(69):354–382, 2004.

2 Boaz Barak, Zeev Dvir, Amir Yehudayoff, and Avi Wigderson. Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 519–528. ACM, 2011.

3 Donald Beaver and Joan Feigenbaum. Hiding instances in multioracle queries. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 37–48. Springer, 1990.

4 Arnab Bhattacharyya, Zeev Dvir, Shubhangi Saraf, and Amir Shpilka. Tight lower bounds for linear 2-query LCCs over finite fields. *Combinatorica*, 36(1):1–36, 2016.

5 Arnab Bhattacharyya and Sivakanth Gopi. Lower bounds for 2-query LCCs over large alphabet. *CoRR*, abs/1611.06980v1, 2016. URL: `http://arxiv.org/abs/1611.06980v1`.

6 Arnab Bhattacharyya and Sivakanth Gopi. Lower bounds for constant query affine-invariant LCCs and LTCs. In *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, pages 12:1–12:17, 2016. `doi:10.4230/LIPIcs.CCC.2016.12`.

7 Manuel Blum and Sampath Kannan. Designing programs that check their work. *J. ACM*, 42(1):269–291, 1995.

8 Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998.

9 Amin Coja-Oghlan, Mikael Onsjö, and Osamu Watanabe. Propagation connectivity of random hypergraphs. *The Electronic Journal of Combinatorics*, 19(1):P17, 2012.

10 Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

11 Richard M. Dudley. Central limit theorems for empirical measures. *The Annals of Probability*, pages 899–929, 1978.

12 Zeev Dvir and Sivakanth Gopi. 2-server PIR with sub-polynomial communication. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 577–584. ACM, 2015.

13 Zeev Dvir, Shubhangi Saraf, and Avi Wigderson. Breaking the quadratic barrier for 3-LCC's over the reals. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 784–793. ACM, 2014.

14 Zeev Dvir, Shubhangi Saraf, and Avi Wigderson. Improved rank bounds for design matrices and a new proof of Kelly's theorem. In *Forum of Mathematics, Sigma*, volume 2, page e4. Cambridge Univ Press, 2014.

15 Zeev Dvir and Amir Shpilka. Locally decodable codes with two queries and polynomial identity testing for depth 3 circuits. *SIAM Journal on Computing*, 36(5):1404–1434, 2007.

16 Oded Goldreich, Howard Karloff, Leonard J. Schulman, and Luca Trevisan. Lower bounds for linear locally decodable codes and private information retrieval. *Computational Complexity*, 15(3):263–296, 2006.

17 Rahul Jain. Towards a classical proof of exponential lower bound for 2-probe smooth codes. *arXiv:cs/0607042*, 2006.

18 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 80–86. ACM, 2000.

19 Neeraj Kayal and Shubhangi Saraf. Blackbox polynomial identity testing for depth 3 circuits. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 198–207. IEEE, 2009.

20 Iordanis Kerenidis and Ronald De Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 106–115. ACM, 2003.

21 Michel Ledoux and Michel Talagrand. *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media, 2013.

22 Richard J. Lipton. Efficient checking of computations. In *Annual Symposium on Theoretical Aspects of Computer Science*, pages 207–215. Springer, 1990.

23 Michael Mitzenmacher and Justin Thaler. Peeling arguments and double hashing. In *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pages 1118–1125. IEEE, 2012.

24 Ryuhei Mori and Osamu Watanabe. Peeling algorithm on random hypergraphs with superlinear number of hyperedges. *arXiv preprint arXiv:1506.00718*, 2015.

25 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

26 Endre Szemerédi. Regular partitions of graphs. In J. C. Bremond, J. C. Fournier, M. Las Vergnas, and D. Sotteau, editors, *Proc. Colloque Internationaux CNRS 260 – Problèmes Combinatoires et Théorie des Graphes*, pages 399–401, 1978.

27 Amelia Taylor. The regularity method for graphs and digraphs. *arXiv preprint arXiv:1406.6531*, 2014.

28 Stephanie Wehner and Ronald De Wolf. Improved lower bounds for locally decodable codes and private information retrieval. In *International Colloquium on Automata, Languages, and Programming*, pages 1424–1436. Springer, 2005.

29 Sergey Yekhanin. Locally decodable codes. In *Computer Science – Theory and Applications*, pages 289–290. Springer, 2011.

A LDCs from LCCs

In this section, we will show that q-query LCCs can be converted into q-query LDCs with only a constant loss in rate and preserving other parameters. Below we define LCCs and LDCs formally.

▶ **Definition 14** (Locally Correctable Code). Let Σ be some finite alphabet and let $\mathcal{C} \subseteq \Sigma^n$ be a set of codewords. $\mathcal{C}$ is called a (q, δ, ε)-LCC if there exists a randomized algorithm $\mathcal{A}$ such that following is true:

1. $\mathcal{A}$ is given oracle access to some $z \in \Sigma^n$ and an input $i \in [n]$. It outputs a symbol in Σ after making at most q queries to z.
2. If $z \in \Sigma^n$ is δ-close to some codeword $c \in \mathcal{C}$ in Hamming distance, then for every $i \in [n]$, $\mathbf{Pr}[\mathcal{A}^z(i) = c_i] \geqslant \frac{1}{2} + \varepsilon$.

It is easy to see that LCCs should have large minimum distance.

▶ **Lemma 15** (Lemma 3.2 in [6]). *If $\mathcal{C} \subseteq \Sigma^n$ is a (q, δ, ε)-LCC, then $\mathcal{C}$ has minimum distance 2δ i.e. every two points in $\mathcal{C}$ are 2δ-far in Hamming distance.*

▶ **Definition 16** (Locally Decodable Code). Let Σ be some finite alphabet and let $\mathcal{C} : \{0,1\}^k \to \Sigma^n$. $\mathcal{C}$ is called a (q, δ, ε)-LDC if there exists a randomized algorithm $\mathcal{A}$ such that following is true:

1. $\mathcal{A}$ is given oracle access to some $z \in \Sigma^n$ and an input $i \in [k]$. It outputs a bit after making at most q queries to z.
2. If $z \in \Sigma^n$ is δ-close to a codeword $\mathcal{C}(x)$ in Hamming distance for some $x \in \{0,1\}^k$, then for every $i \in [k]$, $\mathbf{Pr}[\mathcal{A}^z(i) = x_i] \geqslant \frac{1}{2} + \varepsilon$.

We will need the notion of *VC-dimension* for the reduction.

▶ **Definition 17.** Let $A \subseteq \{0,1\}^n$, then the VC-dimension of A, denoted by $\mathrm{vc}(A)$ is the cardinality of the largest set $I \subseteq [n]$ which is shattered by A i.e. the restriction of A to I, $A|_I = \{0,1\}^I$.

The following lemma due to Dudley([11]) says that if a set $A \subseteq \{0,1\}^n$ has points that are far apart from each other, then it has large VC-dimension.

▶ **Lemma 18** (Theorem 14.12 in [21]). *Let $A \subseteq \{0,1\}^n$ such that for every distinct $x, y \in A$, $\|x-y\|_{\ell_2} \geqslant \varepsilon\sqrt{n}$. Then*

$$\mathrm{vc}(A) \geqslant \Omega\left(\frac{\log|A|}{\log(2/\varepsilon)}\right).$$

We are now ready to prove the reduction from LCCs to LDCs.

▶ **Theorem 19.** *Let $\mathcal{C} \subseteq \Sigma^n$ be a (q,δ,ε)-LCC, then there exists a (q,δ,ε)-LDC $\mathcal{C}' : \{0,1\}^k \to \Sigma^n$ with*

$$k = \Omega\left(\frac{\log|\mathcal{C}|}{\log(1/\delta)}\right).$$

Proof. Wlog let us assume $\Sigma = \{0,1\}^s$. Let $\mathcal{C}_0 : \{0,1\}^s \to \{0,1\}^t$ be an error correcting code with distance δ_0 which is some fixed constant. We can extend $\mathcal{C}_0 : \Sigma^n \to \{0,1\}^{nt}$ as

$$\mathcal{C}_0(z_1,\cdots,z_n) = (\mathcal{C}_0(z_1),\cdots,\mathcal{C}_0(z_n)).$$

By Lemma 15, every two points in $\mathcal{C}$ are 2δ-far in Hamming distance, it is easy to see that in the concatenated code $\mathcal{C}_1 = \mathcal{C}_0 \circ \mathcal{C} \subseteq \{0,1\}^{tn}$ every two points are $2\delta \cdot \delta_0$ far apart in Hamming distance. So every two points in $\mathcal{C}_1$ are separated by $\varepsilon\sqrt{nt}$ distance in ℓ_2 norm where $\varepsilon = \sqrt{2\delta\delta_0}$. So by Lemma 18,

$$\mathrm{vc}(\mathcal{C}_1) \geqslant \Omega\left(\frac{\log|\mathcal{C}_1|}{\log(2/\varepsilon)}\right) = \Omega\left(\frac{\log|\mathcal{C}|}{\log(1/\delta)}\right).$$

Therefore there exists a set $I \subseteq [nt]$ of size $k = \mathrm{vc}(\mathcal{C}_1)$ such that $\mathcal{C}_1|_I = \{0,1\}^I$.

Now define $\mathcal{C}' : \{0,1\}^I \to \Sigma^n$ as follows: $\mathcal{C}'(x) = z$ where $z \in \mathcal{C}$ is chosen such that $\mathcal{C}_0(z)|_I = x$ (if there are many such z, you can choose one arbitrarily). So the image $\mathcal{C}'(\{0,1\}^I) \subseteq \mathcal{C}$. Now we claim that $\mathcal{C}'$ is an q-query LDC. Given a word $r \in \Sigma^n$ which is δ-close to $\mathcal{C}'(x)$, say we want to decode the i^{th} message coordinate x_i. Suppose i belongs to the j^{th} block of $(\{0,1\}^t)^n$ for some $j \in [n]$. The local decoder of $\mathcal{C}'$ will run the local corrector of $\mathcal{C}$ to correct the j^{th} coordinate of r and apply $\mathcal{C}_0$ to find the required bit x_i. So the local decoder for $\mathcal{C}'$ makes at most q queries and the probability that it outputs x_i correctly is at least $1/2 + \varepsilon$. ◀

B Decomposition into expanding subgraphs

The goal of this section is to develop a decomposition lemma that approximately partitions any directed graph into a collection of disjoint expanding subgraphs. We use the following notion of edge expansion:

▶ **Definition 20.** A directed graph $G = (V,E)$ is an *α-edge expander* if for every nonempty $S \subset V$,

$$|E(S, V \setminus S)| \geqslant \alpha|S||V \setminus S|.$$

Here, $E(A,B)$ is the set of edges going from A to B.

We will need the following degree form of Szemerédi regularity lemma which can be derived from the usual form of Szemerédi regularity lemma for directed graphs proved in [1].

▶ **Definition 21.** Let $G = (V, E)$ be a directed graph. We denote the indegree of a vertex $v \in V$ by $\deg^-_G(v)$ and the outdegree by $\deg^+_G(v)$. Given disjoint subsets $A, B \subset V$, the density $d(A, B)$ between A, B is defined as

$$d(A, B) = \frac{E(A, B)}{|A||B|}$$

where $E(A, B)$ is the set of edges going from A to B. We say that (A, B) is ε-regular if for every subsets $A' \subset A$ and $B' \subset B$ such that $|A'| \geqslant \varepsilon|A|$ and $|B'| \geqslant \varepsilon|B|$, $|d(A', B') - d(A, B)| \leqslant \varepsilon$.

Note that the order of A, B is important in the definition of an ε-regular pair.

▶ **Lemma 22** (Szemerédi regularity lemma for directed graphs (see Lemma 39 in [27])). *For every $\varepsilon > 0$, there exists an $M(\varepsilon) > 0$ such that the following is true. Let $G = (V, E)$ be any directed graph on $|V| = n$ vertices and let $0 < d < 1$ be any constant. Then there exists a directed subgraph $G' = (V', E')$ of G and an equipartition of V' into k disjoint parts $V_1, \cdots, V_k$ such that*

1. *$k \leqslant M(\varepsilon)$.*
2. *$|V \setminus V'| \leqslant \varepsilon n$.*
3. *All parts $V_1, \cdots, V_k$ have the same size $m \leqslant \varepsilon n$.*
4. *$\deg^+_{G'}(v) \geqslant \deg^+_G(v) - (d + \varepsilon)n$ for every $v \in V'$.*
5. *$\deg^-_{G'}(v) \geqslant \deg^-_G(v) - (d + \varepsilon)n$ for every $v \in V'$.*
6. *G' doesn't contain edges inside the parts V_i i.e. $E'(V_i, V_i) = \emptyset$ for every i.*
7. *All pairs $G'(V_i, V_j)$ with $i \neq j$ are ε-regular, each with density 0 or at least d.*

The regularity lemma above asserts pseudorandomness in the edges going between parts of the partition. For our application and others, it is more natural to require the edges inside each subgraph to display pseudorandomness. As the proof of our Decomposition Lemma shows, we can obtain this from Lemma 22 with some work.

▶ **Lemma 23** (Decomposition Lemma). *Let $G = (V, E)$ be any directed graph on $|V| = n$ vertices. For $0 < d < 1$ and $0 < \varepsilon < d/6$, there exists a directed subgraph $G' = (V', E')$ and a partition of V' into $U_1, U_2, \cdots, U_K$ where $K \leqslant M(\varepsilon)$ depends only on ε such that:*

1. *$|V \setminus V'| \leqslant 3\varepsilon n$.*
2. *$\deg^+_{G'}(v) \geqslant \deg^+_G(v) - (d + 3\varepsilon)n$ for every $v \in V'$.*
3. *$\deg^-_{G'}(v) \geqslant \deg^-_G(v) - (d + 3\varepsilon)n$ for every $v \in V'$.*
4. *There are no edges from U_i to U_j where $i > j$.*
5. *For $1 \leqslant i \leqslant K$, the induced subgraph $G'(U_i)$ is either empty or is a α-edge expander where $\alpha = \alpha(\varepsilon) > 0$.*

Proof. We will first apply Lemma 22 to G to get a directed subgraph $G''(V'', E'')$ along with a partition of $V'' = V_1 \cup \cdots \cup V_k$ as in the lemma where $k \leqslant M(\varepsilon)$. We know that every pair $G''(V_i, V_j)$ is ε-regular with density 0 or at least d. Let us construct a reduced directed graph $R([k], E_R)$ where $(i, j) \in E_R$ iff $G''(V_i, V_j)$ has density at least d. Now R has a partition into strongly connected components say given by $[k] = S_1 \cup \cdots \cup S_K$ where $K \leqslant M(\varepsilon)$ and $S_1, S_2, \cdots, S_K$ are in topological ordering i.e. there are no edges from S_i to S_j when $i > j$. We will find a large subset $V'_j \subset V_j$ for each of the parts such that $|V_j \setminus V'_j| \leqslant 2\varepsilon|V_j|$ and define $U_i = \cup_{j \in S_i} V'_j$. Our final vertex set will be $V' = \cup_{i=1}^K U_i$ and the graph G' will be the subgraph $G''(V')$. We have

$$|V \setminus V'| \leqslant |V \setminus V''| + \sum_{i=1}^{k} |V_i \setminus V'_i| \leqslant 3\varepsilon n.$$

For every $v \in V'$,

$$\deg^-_{G'}(v) \geqslant \deg^-_{G''}(v) - \sum_{i=1}^{k} |V_i \setminus V'_i| \geqslant \deg^-_G(v) - (d+\varepsilon)n - 2\varepsilon n = \deg^-_G(v) - (d+3\varepsilon)n.$$

Similarly $\deg^+_{G'}(v) \geqslant \deg^+_G(v) - (d+3\varepsilon)n$. Because the components $S_1, \cdots, S_k$ are in topological ordering with respect to the reduced graph R, we cannot have any edges between U_i and U_j where $i > j$.

Now we describe how to find these subsets V'_j where $j \in S_i$ for each of the S_i's and also show the required expansion property. If S_i is a singleton set i.e. $S_i = \{j\}$ for some j, then we just define $V'_j = V_j$. In this case, we will have $U_i = V_j$ and the subgraph $G'(U_i)$ will be empty. If $|S_i| > 1$, the subgraph $R(S_i)$ is strongly connected with at least two vertices. So every vertex $j \in S_i$ has at least one outgoing neighbor and one incoming neighbor in $R(S_i)$; choose one outgoing neighbor and call it $N^+(j)$ and choose one incoming neighbor and call it $N^-(j)$. Let $V'_j \subset V_j$ be the subset of vertices with at least $(d-\varepsilon)|V_{N^+(j)}|$ outgoing neighbors in $V_{N^+(j)}$ and at least $(d-\varepsilon)|V_{N^-(j)}|$ incoming neighbors in $V_{N^-(j)}$. We will now show that $|V_j \setminus V'_j| \leqslant 2\varepsilon|V_j|$. Let $B^+_j \subset V_j$ be the set of vertices with less than $(d-\varepsilon)|V_{N^+(j)}|$ neighbors in $V_{N^+(j)}$. Define $B^-_j \subset V_j$ similarly. We have $V'_j = V_j \setminus (B^+_j \cup B^-_j)$. So it is enough to show $|B^+_j| \leqslant \varepsilon|V_j|$ and $|B^-_j| \leqslant \varepsilon|V_j|$.

Consider the ε-regular pair $(V_j, V_{N^+(j)})$ which has density at least d. The density between B^+_j and $V_{N(j)}$ can be bounded as

$$\frac{|E''(B^+_j, V_{N^+(j)})|}{|B^+_j||V_{N^+(j)}|} < d - \varepsilon \leqslant d(V_j, V_{N^+(j)}) - \varepsilon.$$

By ε-regularity of $G''(V_j, V_{N^+(j)})$, we must have $|B^+_j| \leqslant \varepsilon|V_j|$ as required. Similarly we have $|B^-_j| \leqslant \varepsilon|V_j|$.

Now we need to show that $G'(U_i)$ is an α-edge expander. Let $A \subset U_i$. For $j \in S_i$, define $A_j = A \cap V'_j$ and $\bar{A}_j = V'_j \setminus A$ and let $\bar{A} = U_i \setminus A$. We want to show that $E'(A, \bar{A}) \geqslant \alpha|A||\bar{A}|$ for some constant $\alpha(\varepsilon) > 0$. We have three cases:

Case 1: $\exists j, \ell \in S_i$ such that $|A_j| \geqslant 2\varepsilon|V'_j|$ and $|\bar{A}_\ell| \geqslant 2\varepsilon|V'_\ell|$.

Label vertices of $R(S_i)$ with $\mathcal{A}$ if $|A_j| \geqslant 2\varepsilon|V'_j|$ and also with a label $\bar{\mathcal{A}}$ if $|\bar{A}_j| \geqslant 2\varepsilon|V'_j|$.[8] Every vertex should get at least one of the labels and j has label $\mathcal{A}$ and ℓ has label $\bar{\mathcal{A}}$. Since $|S_i| > 1$, we can assume with out loss of generality that $j \neq \ell$. Since the graph $R(S_i)$ is strongly connected, there is a directed path from j to ℓ. On this path, there must exist two adjacent vertices $p, q \in S_i$ such that p has label $\mathcal{A}$, q has label $\bar{\mathcal{A}}$ and there is an edge from p to q in $R(S_i)$. We have

$$|A_p| \geqslant 2\varepsilon|V'_p| \geqslant 2\varepsilon(1-2\varepsilon)|V_p| \geqslant \varepsilon|V_p|$$

and similarly $|\bar{A}_q| \geqslant \varepsilon|V_q|$. By ε-regularity of $G''(V_p, V_q)$, we can lower the bound the number of edges between A and $\bar{A}$ as follows:

$$|E'(A, \bar{A})| \geqslant |E''(A_p, \bar{A}_q)| \geqslant (d-\varepsilon)|A_p||\bar{A}_q| \geqslant \varepsilon^2(d-\varepsilon)n^2/k^2 \geqslant \alpha_0|A||\bar{A}|$$

where $\alpha_0(\varepsilon) = 5\varepsilon^3/M(\varepsilon)^2$ is some constant depending on ε.

[8] Some vertices can get both labels, but every vertex will get at least one label.

Case 2: For every $j \in S_i$, $|A_j| < 2\varepsilon|V'_j|$.

By averaging there exists some $j \in S_i$ such that $|A_j| \geqslant |A|/|S_i| \geqslant |A|/k$. We know that every vertex in V'_j has at least $(d-\varepsilon)|V_{N^+(j)}|$ out neighbors in $V_{N^+(j)}$, out of these at least

$$(d-\varepsilon)|V_{N^+(j)}| - |V_{N^+(j)} \setminus V'_{N^+(j)}| - |A_{N^+(j)}| \geqslant (d-5\varepsilon)|V_{N^+(j)}|$$

should lie in $\bar{A}_{N^+(j)}$. So we can bound the expansion as follows:

$$|E'(A,\bar{A})| \geqslant |E''(A_j, \bar{A}_{N^+(j)})| \geqslant (d-5\varepsilon)|V_{N^+(j)}||A_j| \geqslant (d-5\varepsilon)\frac{n}{k}\frac{|A|}{k} \geqslant \alpha_1|A||\bar{A}|$$

where $\alpha_1 = \varepsilon/M(\varepsilon)^2$ is some constant depending only on ε.

Case 3: For every $j \in S_i$, $|\bar{A}_j| < 2\varepsilon|V'_j|$.

This is very similar to Case 2. By averaging there exists some $j \in S_i$ such that $|\bar{A}_j| \geqslant |\bar{A}|/|S_i| \geqslant |\bar{A}|/k$. Every vertex in V'_j has at least $(d-\varepsilon)|V_{N^-(j)}|$ incoming neighbors in $V_{N^-(j)}$, out of these at least

$$(d-\varepsilon)|V_{N^-(j)}| - |V_{N^-(j)} \setminus V'_{N^-(j)}| - |\bar{A}_{N^-(j)}| \geqslant (d-5\varepsilon)|V_{N^-(j)}|$$

should lie in $A_{N^-(j)}$. So,

$$|E'(A,\bar{A})| \geqslant |E''(A_{N^-(j)}, \bar{A}_j)| \geqslant (d-5\varepsilon)|V_{N^-(j)}||\bar{A}_j| \geqslant (d-5\varepsilon)\frac{n}{k}\frac{|\bar{A}|}{k} \geqslant \alpha_1|A||\bar{A}|$$

where $\alpha_1 = \varepsilon/M(\varepsilon)^2$.

Finally we can take $\alpha = \min(\alpha_0, \alpha_1)$, to get the required expansion property. ◀

The decomposition lemma allows to give an alternative proof for Claim 7, with worse dependency on τ. To account for that, we restate Claim 7 and replace $O((1/\tau^4)\cdot\log n)$ with $O_\tau(\log n)$.

▶ **Claim 24.** *Let S be a set of size $O_\tau(\log n)$ such that $R(S) = T_1$. Then, S can be extended by at most $O_\tau(\log n)$ elements, such that $R(S) = V$.*

Proof. Let $\{\mathcal{M}_v : v \in T_2\}$ be the matchings obtained from Lemma 3, we know that $|\mathcal{M}_v| \geqslant \frac{\tau}{4}n$ for each $v \in T_2$. We will construct a directed graph $G(V,E)$ where $V = [n]$ and E is defined as follows. For every $v \in T_2 \setminus R(S)$ and every edge $\{i,j\} \in \mathcal{M}_v$, add directed edges $(i,v),(j,v)$ to E. Thus there is a natural pairing among the directed edges of G, we will call (j,v) the *pairing edge* of (i,v) and vice versa. $\{i,j\}$ is called the *matching edge* corresponding to the pair $(i,v),(j,v)$. Since each matching $\mathcal{M}_v$ has size $\geqslant \tau n/4$, we have $\deg^-_G(v) \geqslant \delta n$ where $\delta := \tau/2$ for every $v \in T_2 \setminus R(S) = V \setminus R(S)$.

We now apply Lemma 23 to get a subgraph $G' = (V',E')$ as described in the lemma where we will choose $\varepsilon = \delta/100$ and $d = \delta/10$. Let $V' = U_1 \cup \cdots \cup U_K$ be the partition of G' as described in the lemma where $K \leqslant M(\delta)$. Let $V_0 = [n] \setminus V'$ be the remaining vertices, we have $|V_0| \leqslant 3\varepsilon n$. Each vertex $v \in V' \cap (T_2 \setminus R(S))$ has $\deg^-_{G'}(v) \geqslant (\delta - d - 3\varepsilon)n$. We also know that each sub-graph $G'(U_i)$ is either empty or is an α-edge expander for some constant $\alpha(\varepsilon) > 0$.

Note that S already has $O_\tau(\log n)$ vertices. We will now grow the set S of coordinates queried by $\mathcal{P}$ iteratively, adding one at a time. Algorithm 1 gives the procedure for growing the set S.

We will finish the analysis in a series of claims. Let us start with a simple claim about properties of $R(S)$.

Algorithm 1 Algorithm for growing S

```
for $i = 1$ to $K$ do
  Intialization: Pick one vertex from $U_i$ and add it to $S$.
  while $U_i \nsubseteq R(S)$ do
    Pick any $v \in V \setminus R(S)$ such that adding it to $S$ will add the maximum number of vertices in $U_i \setminus R(S)$ to $R(S)$.
  end while
end for
```

▶ **Claim 25.** *$R(S)$ has the following properties:*

1. *If $i, j \in R(S)$ and $(i, j) \in \mathcal{M}_k$ then $k \in R(S)$.*
2. *For every edge $(i, k) \in E(R(S), V \setminus R(S))$, there is a unique $j \in V \setminus R(S)$ such that $(i, j) \in \mathcal{M}_k$.*

Proof. (1) We can recover c_i, c_j from $c|_S$ and then use them to recover c_k since by Lemma 3, there exists an algorithm $\mathcal{R}^k_{i,j}$ such that for every $c \in \mathcal{C}$, $\mathcal{R}^k_{i,j}(c_i, c_j) = c_k$.
(2) Let (j, k) be the pairing edge of (i, k) so that $(i, j) \in \mathcal{M}_k$. Now j cannot be in $R(S)$ because of (1). ◀

Algorithm 1 should terminate, since $|U_i \cap R(S)|$ increases by at least one in every iteration of the while loop. At the end of the procedure we clearly have $V' = U_1 \cup \cdots \cup U_K \subset R(S)$. In fact, we can claim that at the end of the procedure $R(S) = V$ i.e. we can recover all the coordinates of c from $c|_S$.

▶ **Claim 26.** *After Algorithm 1 terminates, $R(S) = V = [n]$.*

Proof. After Algorithm 1 terminates, we have $V' \subset R(S)$. Now we are left with $V_0 = V \setminus V'$ where we know that $|V_0| \leqslant 3\varepsilon n$. Now if $w \in V_0 \setminus R(S)$ then $w \in T_2 \setminus R(S)$ since $T_1 \subset R(S)$. Therefore $\deg^-_G(w) \geqslant \delta n$. So there must be $\delta n - |V_0| \geqslant (\delta - 3\varepsilon)n$ incoming edges from V' to w. So two of these incoming edges must from a pair and so we have $w \in R(S)$ by part (1) of Claim 25. Therefore $V_0 \subset R(S)$ as well. ◀

▶ **Claim 27.** *Algorithm 1 terminates after $O_\delta(\log n)$ rounds.*

Proof. We just need to show that the while loop runs for $O_\delta(\log n)$ rounds for each $i \in [K]$ since the outer for loop runs for K times where $K \leqslant M(\delta)$. There are two cases:

Case 1: The subgraph $G'(U_i)$ is empty.

In this case, we will show that U_i must already be contained in $R(S)$. Suppose not, let $w \in U_i \setminus R(S)$, we have $\deg^-_{G'}(w) \geqslant (\delta - d - 3\varepsilon)n$. Moreover, all of these incoming edges come from $U_1, \cdots, U_{i-1}$ (note that this means $i > 1$ for this case to happen). Therefore there must be two incoming edges from $U_1 \cup \cdots \cup U_{i-1}$ which form a pair i.e. there exists $u, v \in U_1 \cup \cdots \cup U_{i-1}$ such that $(u, v) \in \mathcal{M}_w$. So by part (1) of Claim 25, $w \in R(S)$. This is a contradiction.

Case 2: The subgraph $G'(U_i)$ is an α-edge expander.

If $U_i \nsubseteq R(S)$, we will show that after the end of the iteration $t_i := |R(S) \cap U_i|$ increases by a factor of $(1 + \varepsilon\alpha)$. This will prove the required claim because t_i is upper bounded by n.

We first claim that $|U_i \setminus R(S)| \geqslant \varepsilon n$. Suppose this is not true i.e. $|U_i \setminus R(S)| \leqslant \varepsilon n$. Let $w \in U_i \setminus R(S)$. We know that w has $\deg^-_{G'}(w) \geqslant (\delta - d - 3\varepsilon)n$ incoming edges in G'. Since no edges come from U_j for $j > i$, at least $(\delta - d - 3\varepsilon)n - |U_i \setminus R(S)| \geqslant (\delta - d - 4\varepsilon)n$

of them come from $U_1 \cup \cdots \cup U_{i-1} \cup (U_i \cap R(S)) \subset R(S)$. Therefore two of the incoming edges must form a pair and so $w \in R(S)$ which is a contradiction.
Since $G'(U_i)$ is an α-edge expander, we have

$$E(U_i \cap R(S), U_i \setminus R(S)) \geqslant \alpha t_i |U_i \setminus R(S)| \geqslant \alpha \varepsilon t_i n.$$

By part (2) of Claim 25, each edge from $U_i \cap R(S)$ to $U_i \setminus R(S)$ corresponds to a matching edge between $U_i \cap R(S)$ and $V \setminus R(S)$ and it belongs to a matching which corresponds to a vertex in $U_i \setminus R(S)$. Therefore there are at least $\alpha\varepsilon t_i n$ matching edges between $U_i \cap R(S)$ and $V \setminus R(S)$ which belong to $\cup_{w \in U_i \setminus R(S)} \mathcal{M}_w$; by averaging there exists $v \in V \setminus R(S)$ which is incident to $\alpha\varepsilon t_i n / |V \setminus R(S)| \geqslant \alpha\varepsilon t_i$ of these matching edges. So adding this v to S will add $\alpha\varepsilon t_i$ new vertices of $U_i \setminus R(S)$ to $R(S)$, increasing t_i by a factor of $(1 + \alpha\varepsilon)$.

◀

◀

Sum-of-Squares Certificates for Maxima of Random Tensors on the Sphere

Vijay Bhattiprolu[*1], **Venkatesan Guruswami**[†2], **and Euiwoong Lee**[‡3]

1 **Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA**
vpb@cs.cmu.edu

2 **Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA**
guruswami@cmu.edu

3 **Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA**
euiwoonl@cs.cmu.edu

Abstract

For an n-variate order-d tensor $\mathcal{A}$, define $\mathcal{A}_{\max} := \sup_{\|x\|_2=1} \langle \mathcal{A}, x^{\otimes d} \rangle$ to be the maximum value taken by the tensor on the unit sphere. It is known that for a random tensor with i.i.d. ±1 entries, $\mathcal{A}_{\max} \lesssim \sqrt{n \cdot d \cdot \log d}$ w.h.p. We study the problem of efficiently certifying upper bounds on $\mathcal{A}_{\max}$ via the natural relaxation from the Sum of Squares (SoS) hierarchy. Our results include:

- When $\mathcal{A}$ is a random order-q tensor, we prove that q levels of SoS certifies an upper bound B on $\mathcal{A}_{\max}$ that satisfies

  $$B \quad \leq \quad \mathcal{A}_{\max} \cdot \left(\frac{n}{q^{\,1-o(1)}} \right)^{q/4-1/2} \qquad \text{w.h.p.}$$

  Our upper bound improves a result of Montanari and Richard (NIPS 2014) when q is large.
- We show the above bound is the best possible up to lower order terms, namely the optimum of the level-q SoS relaxation is at least

  $$\mathcal{A}_{\max} \cdot \left(\frac{n}{q^{\,1+o(1)}} \right)^{q/4-1/2} .$$

- When $\mathcal{A}$ is a random order-d tensor, we prove that q levels of SoS certifies an upper bound B on $\mathcal{A}_{\max}$ that satisfies

  $$B \;\leq\; \mathcal{A}_{\max} \cdot \left(\frac{\widetilde{O}(n)}{q} \right)^{d/4-1/2} \qquad \text{w.h.p.}$$

  For growing q, this improves upon the bound certified by constant levels of SoS. This answers in part, a question posed by Hopkins, Shi, and Steurer (COLT 2015), who gave the tight characterization for constant levels of SoS.

1998 ACM Subject Classification G.1.6 Optimization, F.2.1 Numerical Algorithms and Problems

Keywords and phrases Sum-of-Squares; Optimization over Sphere; Random Polynomials

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.31

[*] Supported by NSF grants CCF-1422045 and CCF-1526092.
[†] Research supported in part by NSF grants CCF-1526092 and CCF-1563742.
[‡] Supported by a Samsung Fellowship, a Simons Award for Graduate Students in Theoretical Computer Science, and NSF CCF-1526092.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 31; pp. 31:1–31:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

It is a well-known fact from random matrix theory that for an $n \times n$ matrix M whose entries are i.i.d. Rademacher or standard normal random variables, the maximum value $x^T Mx$ taken by the associated quadratic form on the unit sphere $\|x\|_2 = 1$, is $\Theta(\sqrt{n})$ with high probability. Further, this maximum value can be computed efficiently for any matrix, as it equals the largest eigenvalue of $(M + M^T)/2$, so one can also efficiently certify that the maximum of a random quadratic form is at most $O(\sqrt{n})$.

This paper is motivated by the problem of analogous question for tensors. Namely, given a random order-d tensor $\mathcal{A}$ who entries are i.i.d. random ± entries, we would like to certify an upper bound on the maximum value $\mathcal{A}_{\max} := \max_{\|x\|=1}\langle \mathcal{A}, x^{\otimes d}\rangle$ taken by the tensor on the unit sphere. This value is at most $O_d(\sqrt{n})$ with high probability [18]. However, for $d \geq 3$, computing $\mathcal{A}_{\max}$ for a d-tensor $\mathcal{A}$ is NP-hard, and it is likely that the problem is also very hard to approximate. Assuming the Exponential Time Hypothesis, Barak et al. [1] proved that computing $2 \to 4$ norm of a matrix, a special case of computing the norm of a 4-tensor, is hard to approximate within a factor $\exp(\log^{1/2-\epsilon}(n))$ for any $\epsilon > 0$.

Our goal is to certify an *approximate* upper bound on $\mathcal{A}_{\max}$ is not too far from the true value. Specifically, we seek an estimate $B(\mathcal{A})$ which always upper bounds $\mathcal{A}_{\max}$, and with high probability is as close to $O_d(\sqrt{n})$ as possible for a random $\mathcal{A}$.

In addition to its intrinsic interest, the problem of maximizing tensors and closely related tasks of computing tensor norms, has connections to diverse topics, such as quantum information theory [7, 2], the Small Set Expansion Hypothesis (SSEH) and the Unique Games Conjecture (UGC) (via $2 \to 4$ norm, see [1, 2]), refuting random CSPs [16], tensor decomposition [3, 10], tensor PCA [15, 12], and planted clique (via the parity tensor, see [9, 8]). Many of these applications are of considerable interest in the 2^{n^ϵ}-runtime regime.

A natural approach to tackle the above problem is through the *Sum of Squares* (SoS) semidefinite programming relaxations. There are several ways to represent a tensor $\mathcal{A} \in \mathbb{R}^{[n]^d}$ (assume d is even) in matrix form as $M \in \mathbb{R}^{[n]^{d/2} \times [n]^{d/2}}$ so that $\langle \mathcal{A}, x^{\otimes d}\rangle = (x^{\otimes d/2})^T M x^{\otimes d/2}$ for all $x \in \mathbb{R}^n$. The largest eigenvalue $\lambda_{\max}(M)$ of any such matrix representation M serves as an (efficiently computable) upper bound on $\mathcal{A}_{\max}$. The basic SoS relaxation looks for the best matrix representation, i.e., the one minimizing $\lambda_{\max}(M)$, among all possible representations of the tensor $\mathcal{A}$. This can be expressed as a semidefinite program, and also has a natural dual view in terms of pseudo-expectations or moment matrices (see Section 2.2).

The SoS hierarchy offers a sequence of relaxations, parameterized by the *level* q, with larger q giving a (potentially) tighter relaxation. In our context, this amounts to optimizing over matrix representations of $\mathcal{A}^{q/d}$ (we assume q is divisible by $2d$); in the dual view, this involves optimizing over pseudo-expectations for polynomials of degree up to q (as opposed to degree d for the basic relaxation). The level-q relaxation can be solved in $n^{O(q)}$ time by solving the associated semidefinite program. The SoS hierarchy thus presents a trade-off between approximation guarantee and runtime, with larger levels giving more accurate estimates at the expense of higher complexity.

This work is concerned with both positive and negative results on the efficacy of the SoS hierarchy to approximately certify the maxima of random tensors. We now turn to stating our results formally.

1.1 Our Results

For an order-q tensor $\mathcal{A} \in (\mathbb{R}^n)^{\otimes d}$, the polynomial $\mathcal{A}(x)$ and its maximum on the sphere $\mathcal{A}_{\max}$ are defined as

$$\mathcal{A}(x) := \langle \mathcal{A}, x^{\otimes d} \rangle \qquad \mathcal{A}_{\max} := \sup_{\|x\|=1} \mathcal{A}(x).$$

When the entries of $\mathcal{A}$ are i.i.d. Rademacher random variables (or i.i.d. Gaussians), it is known that $\mathcal{A}_{\max} \lesssim \sqrt{n \cdot d \cdot \log d}$ (see [18]). We will also use, for a polynomial g, $g_{\max}$ to denote $\sup_{\|x\|=1} g(x)$.

SoS degree = Polynomial Degree

We study the performance of degree-q SoS on random tensors of order-q. The formal definition and basic properties of SoS relaxations are presented in Section 2.2.

▶ **Theorem 1.** *For any even $q \leq n$, let $\mathcal{A} \in (\mathbb{R}^n)^{\otimes q}$ be a q-tensor with independent, Rademacher entries. With high probability, the value B of the degree-q SoS relaxation of $\mathcal{A}_{\max}$ satisfies*

$$2^{-O(q)} \cdot \left(\frac{n}{q}\right)^{q/4-1/2} \leq \frac{B}{\mathcal{A}_{\max}} \leq 2^{O(q)} \cdot \left(\frac{n}{q}\right)^{q/4-1/2}.$$

This improves upon the $O(n^{q/4})$ upper bound by Montanari and Richard [15].

SoS Degree ≫ Polynomial Degree

▶ **Theorem 2.** *Let $\mathcal{A} \in (\mathbb{R}^n)^{\otimes d}$ be a d-tensor with independent, Rademacher entries. Then for any even q satisfying $d \leq q \leq n$, with high probability, the degree-q SoS certifies an upper bound B on $\mathcal{A}_{\max}$ where w.h.p.,*

$$\frac{B}{\mathcal{A}_{\max}} \leq \left(\frac{\widetilde{O}(n)}{q}\right)^{d/4-1/2}$$

▶ Remark. Combining our upper bounds with the work of [12] would yield improved tensor-PCA guarantees on higher levels of SoS.

▶ Remark. Raghavendra, Rao, and Schramm [16] have independently and concurrently obtained similar (but weaker) results to Theorem 2 for random degree-d polynomials. Specifically, their upper bounds appear to require the assumption that the SoS level q must be less than $n^{1/(3d^2)}$ (our result only assumes $q \leq n$). Further, they certify an upper bound that matches Theorem 2 only when $q \leq 2^{\sqrt{\log n}}$.

1.2 Related Work

Upper Bounds

Montanari and Richard [15] presented a $n^{O(d)}$-time algorithm that can certify that the optimal value of $\mathcal{A}_{\max}$ for a random d-tensor is at most $O(n^{\frac{\lceil d/2 \rceil}{2}})$ with high probability. Hopkins, Shi, and Steurer [12] improved it to $O(n^{\frac{d}{4}})$ with the same running time. They also asked how many levels of SoS are required to certify a bound of $n^{3/4-\delta}$ for $d = 3$.

Our analysis asymptotically improves the aforementioned bound when q is growing with n, and we prove an essentially matching lower bound (but only for the case $q = d$). Secondly, we consider the case when d is fixed, and give improved results for the performance of degree-q SoS (for large q), thus answering in part, a question posed by Hopkins, Shi and Steurer [12].

Raghavendra, Rao, and Schramm [16] also prove results analogous to Theorem 2 for the case of *sparse* random polynomials (a model we do not consider in this work, and which appears to pose additional technical difficulties). This implied upper bounds for refuting random instances of constraint satisfaction problems using higher levels of the SoS hierarchy, which were shown to be tight via matching SoS lower bounds in [13].

Lower Bounds

While we only give lower bounds for the case of $q = d$, subsequent to our work, Hopkins et al. [11] proved the following theorem, which gives lower bounds for the case of $q \gg d$:

▶ **Theorem 3.** *Let f be a degree-d polynomial with i.i.d. gaussian coefficients. If there is some constant $\epsilon > 0$ such that $q \geq n^{\epsilon}$, then with high probability over f, the optimum of the level-q SoS relaxation of $f_{\max}$ is at least*

$$f_{\max} \cdot \Omega_d\Big((n/q^{O(1)})^{d/4-1/2}\Big) \ .$$

Note that this almost matches our upper bounds from Theorem 2, modulo the exponent of q. For this same reason, the above result does not completely recover our lower bound in Theorem 1 for the special case of $q = d$.

Results for worst-case tensors

It is proved in [5] that the q-level SoS gives an $(O(n)/q)^{d/2-1}$ approximation to $\|\mathcal{A}\|_2$ in the case of arbitrary d-tensors and an $(O(n)/q)^{d/4-1/2}$ approximation to $\mathcal{A}_{\max}$ in the case of d-tensors with non-negative entries (for technical reasons one can only approximate $\|\mathcal{A}\|_2 = \max\{|\mathcal{A}_{\max}|, |\mathcal{A}_{\min}|\}$ in the former case).

It is interesting to note that the approximation factor in the case of non-negative tensors matches the approximation factor (upto polylogs) we achieve in the random case. Additionally, the gap given by Theorem 1 for the case of random tensors provides the best degree-q SoS gap for the problem of approximating the 2-norm of arbitrary q-tensors. Hardness results for the arbitrary tensor 2-norm problem is an important pursuit due to its connection to various problems for which subexponential algorithms are of interest.

1.3 Organization

We begin by setting some important notation concerning SoS matrices, and describe some basic preliminaries about the SoS hierarchy in Section 2. We touch upon the main technical ingredients driving our work, and give an overview of the proof of Theorem 2 and the lower bound in Theorem 1 in Section 3. We present the proof of Theorem 2 for the case of even d in Section 4, with the more tricky odd d case handled in the full version of our paper [6]. The lower bound on the value of SoS-hierarchy claimed in Theorem 1 is proved in Section 5, and the upper bound in Theorem 1 also follows based on some techniques in that section.

2 Notation and Preliminaries

Multi-index and Multiset

A multi-index is defined as a sequence $\alpha \in \mathbb{N}^n$. We use $|\alpha|$ to denote $\sum_{i=1}^{n} \alpha_i$ and $\mathbb{N}^n_d$ (resp. $\mathbb{N}^n_{\leq d}$) to denote the set of all multi-indices α with $|\alpha| = d$ (resp. $|\alpha| \leq d$). We use $\mathbf{1}$ to denote the multi-index 1^n. Thus, a homogeneous polynomial f of degree d can be expressed in terms of its coefficients as

$$f(x) \;=\; \sum_{\alpha \in \mathbb{N}^n_d} f_\alpha \cdot x^\alpha,$$

where x^α is used to denote the monomial corresponding to α. In general, with the exception of absolute-value, any scalar function/operation when applied to vectors/multi-indices, returns the vector obtained by applying the function/operation entry-wise.

2.1 Matrices

For $k \in \mathbb{N}$, we will consider $[n]^k \times [n]^k$ matrices M with real entries. All matrices considered in this paper should be taken to be symmetric (unless otherwise stated). We index entries of the matrix M as $M[I, J]$ by tuples $I, J \in [n]^k$. $\oplus$ denotes tuple-concatenation.

A tuple $I = (i_1, \ldots, i_k)$ naturally corresponds to a multi-index $\alpha(I) \in \mathbb{N}^n_k$ with $|\alpha(I)| = k$, i.e. $\alpha(I)_j = |\{\ell \mid i_\ell = j\}|$. For a tuple $I \in [n]^k$, we define $\mathcal{O}(I)$ the set of all tuples J which correspond to the same multi-index i.e., $\alpha(I) = \alpha(J)$. Thus, any multi-index $\alpha \in \mathbb{N}^n_k$, corresponds to an equivalence class in $[n]^k$. We also use $\mathcal{O}(\alpha)$ to denote the class of all tuples corresponding to α.

Note that a matrix of the form $\left(x^{\otimes k}\right)\left(x^{\otimes k}\right)^T$ has many additional symmetries, which are also present in solutions to programs given by the SoS hierarchy. To capture this, consider the following definition:

▶ **Definition 4** (SoS-Symmetry). A matrix M which satisfies $\mathsf{M}[I, J] = \mathsf{M}[K, L]$ whenever $\alpha(I) + \alpha(J) = \alpha(K) + \alpha(L)$ is referred to as SoS-symmetric.

▶ **Definition 5** (Matrix-Representation). For a homogeneous degree-t (t even) polynomial g, we say a matrix $\mathrm{M}_g \in \mathbb{R}^{[n]^{t/2} \times [n]^{t/2}}$ is a degree-t matrix representation of g if for all x, $g(x) = (x^{\otimes t/2})^T\ \mathrm{M}_g\ x^{\otimes t/2}$. (We note here that every homogeneous polynomial has a unique SoS-Symmetric matrix representation.)

Note that $\lambda_{\max}(\mathrm{M}_g)$ is an upper bound on $g_{\max}$. This prompts the following relaxation of $g_{\max}$ that is closely related to the final SoS relaxation used in our upper bounds:

▶ **Definition 6.** For a homogeneous degree-t (t even) polynomial g, define

$$\Lambda(g) \;:=\; \inf\left\{\lambda_{\max}(M_g) \;\middle|\; M_g \text{ represents } g\right\}.$$

As we will see shortly, $\Lambda(g)$ is the dual of a natural SoS relaxation of $g_{\max}$.

2.2 SoS Hierarchy

Let $\mathbb{R}[x]_{\leq q}$ be the vector space of polynomials with real coefficients in variables $x = (x_1, \ldots, x_n)$, of degree at most q. For an even integer q, the degree-q pseudo-expectation operator is a linear operator $\widetilde{\mathbf{E}} : \mathbb{R}[x]_{\leq q} \mapsto \mathbb{R}$ such that

1. $\widetilde{\mathbf{E}}\,[1] = 1$ for the constant polynomial 1.
2. $\widetilde{\mathbf{E}}\,[p_1 + p_2] = \widetilde{\mathbf{E}}\,[p_1] + \widetilde{\mathbf{E}}\,[p_2]$ for any polynomials $p_1, p_2 \in \mathbb{R}[x]_{\leq q}$.
3. $\widetilde{\mathbf{E}}\left[p^2\right] \geq 0$ for any polynomial $p \in \mathbb{R}[x]_{\leq q/2}$.

The pseudo-expectation operator $\widetilde{\mathbf{E}}$ can be completely described by the ***moment matrix*** (while x is a column vector, we abuse notation and let $(1,x)$ denote the column vector $(1,x_1,\ldots,x_n)^T$)

$$\overline{\mathsf{X}} := \widetilde{\mathbf{E}}\left[(1,x)^{\otimes q/2}\ ((1,x)^{\otimes q/2})^T\right] . \tag{2.1}$$

Moreover, the condition $\widetilde{\mathbf{E}}\left[p^2\right] \geq 0$ for all $p \in \mathbb{R}[x]_{\leq q/2}$ can be shown to be equivalent to $\overline{\mathsf{X}} \succeq 0$.

Constrained Pseudoexpectations

For a system of polynomial constraints

$$C = \{f_1 = 0, \ldots, f_m = 0, g_1 \geq 0, \ldots, g_r \geq 0\},$$

we say $\widetilde{\mathbf{E}}_C$ is a pseudoexpectation operator respecting C, if in addition to the above conditions, it also satisfies

1. $\widetilde{\mathbf{E}}_C[p \cdot f_i] = 0$, $\forall i \in [m]$ and $\forall p$ such that $\deg(p \cdot f_i) \leq q$.
2. $\widetilde{\mathbf{E}}_C\left[p^2 \cdot \prod_{i\in S} g_i\right] \geq 0$, $\forall S \subseteq [r]$ and $\forall p$ such that $\deg(p^2 \cdot \prod_{i\in S} g_i) \leq q$.

It is well-known that such constrained pseudoexpectation operators can be described as solutions to semidefinite programs of size $n^{O(q)}$ [4, 14]. This hierarchy of semidefinite programs for increasing q is known as the SoS hierarchy.

Additional Facts about SoS

We shall record here some well-known facts about SoS that come in handy later.

▶ **Claim 7.** *For polynomials p_1, p_2, let $p_1 \succeq p_2$ denote that $p_1 - p_2$ is a sum of squares. It is easy to verify that if p_1, p_2 are homogeneous degree d polynomials and there exist matrix representations M_{p_1} and M_{p_2} of p_1 and p_2 respectively, such that $M_{p_1} - M_{p_2} \succeq 0$, then $p_1 - p_2 \succeq 0$.*

▶ **Claim 8** (Pseudo-Cauchy-Schwarz [2])**.** $\widetilde{\mathbf{E}}\left[p_1 p_2\right] \leq (\widetilde{\mathbf{E}}\left[p_1^2\right] \widetilde{\mathbf{E}}\left[p_2^2\right])^{1/2}$ *for any p_1, p_2 of degree at most $q/2$.*

SoS Relaxations for $\mathcal{A}_{\max}$

Given an order-q tensor $\mathcal{A}$, our degree-q SoS relaxation for $\mathcal{A}_{\max}$ which we will henceforth denote by $\mathsf{SoS}_q(\mathcal{A}(x))$ is given by,

$$\begin{array}{rr} \text{maximize} & \widetilde{\mathbf{E}}_C[\mathcal{A}(x)] \\ \text{subject to :} & \widetilde{\mathbf{E}}_C \text{ is a degree-}q \\ & \text{pseudoexpectation} \\ \multicolumn{2}{r}{\widetilde{\mathbf{E}}_C \text{ respects } C \equiv \{\|x\|_2^q = 1\}} \end{array}$$

Assuming q is divisible by $2d$, we make an observation that is useful in our upper bounds:

$$\mathcal{A}_{\max} \ \leq\ \mathsf{SoS}_q(\mathcal{A}(x)) \ \leq\ \mathsf{SoS}_q\Big(\mathcal{A}(x)^{\,q/d}\Big)^{d/q} \ =\ \Lambda\Big(\mathcal{A}(x)^{\,q/d}\Big)^{d/q} \tag{2.2}$$

where the second inequality follows from Pseudo-Cauchy-Scwarz, and the equality follows from well known strong duality of the following programs (specifically, take $g(x) := \mathcal{A}(x)^{\,q/d}$):[1]

[1] Compared to (2.1), the primal formulation here uses a *homogeneous* moment matrix or pseudo-expectation operator, defined for polynomials of degree exactly q.

Dual

$$\Lambda(g) \;:=\; \inf\left\{\lambda_{\max}(M_g) \;\middle|\; M_g \text{ represents } g\right\}$$

Primal I

$$\begin{array}{lr} \text{maximize} & \langle \mathsf{M}_g, \mathsf{X}\rangle \\ \text{subject to :} & \mathbf{Tr}(\mathsf{X}) = 1 \\ & \mathsf{X} \text{ is SoS symmetric} \\ & \mathsf{X} \succeq 0 \end{array}$$

Primal II

$$\begin{array}{lr} \text{maximize} & \widetilde{\mathbf{E}}_C[g] \\ \text{subject to :} & \widetilde{\mathbf{E}}_C \text{ is a degree-}q \\ & \text{pseudoexpectation} \\ & \widetilde{\mathbf{E}}_C \text{ respects } C \equiv \left\{\|x\|_2^q = 1\right\} \end{array}$$

Figure 2.1 Duals of $\Lambda(g)$ for the degree-q homogeneous polynomial g.

Note

In the rest of the paper, we will drop the subscript C of the pseudo-expectation operator since throughout this work, we only assume the hypersphere constraint.

3 Overview of our Methods

We now give a high level view of the two broad techniques driving this work, followed by a more detailed overview of the proofs.

Higher Order Mass-Shifting

Our approach to upper bounds on a random low degree (say d) polynomial f, is through exhibiting a matrix representation of $f^{q/d}$ that has small operator norm. Such approaches had been used previously for low-degree SoS upper bounds. However when the SoS degree is constant, the set of SoS symmetric positions is also a constant and the usual approach is to shift all the mass towards the diagonal which is of little consequence when the SoS-degree is low. In contrast, when the SoS-degree is large, many non-trivial issues arise when shifting mass across SoS-symmetric positions, as there are many permutations with very large operator norm. In our setting, mass-shifting approaches like symmetrizing and diagonal-shifting fail quite spectacularly to provide good upper bounds. For our upper bounds, we crucially exploit the existence of "good permutations", and moreover that there are $q^q \cdot 2^{-O(q)}$ such good permutations. On averaging the representations corresponding to these good permutations, we obtain a matrix that admits similar spectral properties to those of a matrix with i.i.d. entries, and with much lower variance (in most of the entries) compared to the naive representations.

Square Moments of Wigner Semicircle Distribution

Often when one is giving SoS lower bounds, one has a linear functional that is not necessarily PSD and a natural approach is to fix it by adding a pseudo-expectation operator with large value on square polynomials (under some normalization). Finding such operators however, is quite a non-trivial task when the SoS-degree is growing. We show that if $x_1, \ldots, x_n$ are independently drawn from the Wigner semicircle distribution, then for any polynomial p

of any degree, $\mathbf{E}\left[p^2\right]$ is large (with respect to the degree and coefficients of p). Our proof crucially relies on knowledge of the Cholesky decomposition of the moment matrix of the univariate Wigner distribution. This tool was useful to us in giving tight q-tensor lower bounds, and we believe it to be generally useful for high degree SoS lower bounds.

3.1 Overview of Upper Bound Proofs

For even d, let $\mathcal{A} \in \mathbb{R}^{[n]^d}$ be a d-tensor with i.i.d. ±1 entries and let $A \in \mathbb{R}^{[n]^{d/2} \times [n]^{d/2}}$ be the matrix flattening of $\mathcal{A}$, i.e., $A[I, J] = \mathcal{A}[I \oplus J]$ (recall that $\oplus$ denotes tuple concatenation). Also let $f(x) := \mathcal{A}(x) = \langle \mathcal{A}, x^{\otimes d} \rangle$. It is well known that $f_{\max} \le O(\sqrt{n \cdot d \cdot \log d})$ with high probability [18]. For such a polynomial f and any q divisible by d, in order to establish Theorem 2, by Eq. (2.2) it is sufficient to prove that with high probability,

$$\left(\Lambda\left(f^{q/d}\right)\right)^{d/q} \;\le\; \widetilde{O}\left(\frac{n}{q^{1-2/d}}\right)^{d/4} \;=\; \widetilde{O}\left(\frac{n}{q}\right)^{d/4-1/2} \cdot f_{\max}.$$

We give an overview of the proof. Let $d = 4$ for the sake of clarity of exposition. To prove an upper bound on $\Lambda\left(f^{q/4}\right)$ using degree-q SoS (assume q is a multiple of 4), we define a suitable matrix representation $M := M_{f^{q/4}} \in \mathbb{R}^{[n]^{q/2} \times [n]^{q/2}}$ of $f^{q/4}$ and bound $\|M\|_2$. Since $\Lambda(f) \le (\|M\|_2)^{q/4}$ for any representation M, a good upper bound on $\|M\|_2$ certifies that $\Lambda(f)$ is small.

One of the intuitive reasons taking a high power gives a better bound on the spectral norm is that this creates more entries of the matrix that correspond to the same monomial, and distributing the coefficient of this monomial equally among the corresponding entries reduces variance (i.e., $\mathbf{Var}[X]$ is less than $k \cdot \mathbf{Var}[X/k]$ for $k > 1$). In this regard, the most natural representation M of $f^{q/4}$ is the *complete symmetrization*.

$$\begin{aligned}
&M_c[(i_1, \ldots, i_{q/2}), (i_{q/2+1}, \ldots, i_q)] \\
&= \frac{1}{q!} \cdot \sum_{\pi \in \mathbb{S}_q} A^{\otimes q/4}[(i_{\pi(1)}, \ldots, i_{\pi(q/2)}), (i_{\pi(q/2+1)}, \ldots, i_{\pi(q)})] \\
&= \frac{1}{q!} \cdot \sum_{\pi \in \mathbb{S}_q} \prod_{j=1}^{q/4} A[(i_{\pi(2j-1)}, i_{\pi(2j)}), (i_{\pi(q/2+2j-1)}, i_{\pi(q/2+2j)})].
\end{aligned}$$

However, $\|M_c\|_2$ turns out to be much larger than $\Lambda(f)$, even when $q = 8$. One intuitive explanation is that M_c, as a $n^4 \times n^4$ matrix, contains a copy of $\mathbf{Vec}(A)\,\mathbf{Vec}(A)^T$, where $\mathbf{Vec}(A) \in \mathbb{R}^{[n]^4}$ is the vector with $\mathbf{Vec}(A)\,[i_1, i_2, i_3, i_4] = A[(i_1, i_2), (i_3, i_4)]$. Then $\mathbf{Vec}(A)$ is a vector that witnesses $\|M_c\|_2 \ge \Omega(n^2)$, regardless of the randomness of f. Our final representation[2] is the following *row-column independent symmetrization* that simultaneously respects the spectral structure of a random matrix A and reduces the variance. Our M is given by

$$\begin{aligned}
&M[(i_1, \ldots, i_{q/2}), (j_1, \ldots, j_{q/2})] \\
&= \frac{1}{(q/2)!^2} \cdot \sum_{\pi, \sigma \in \mathbb{S}_{q/2}} A^{\otimes q/4}[(i_{\pi(1)}, \ldots, i_{\pi(q/2)}), (j_{\sigma(1)}, \ldots, j_{\sigma(q/2)})] \\
&= \frac{1}{(q/2)!^2} \cdot \sum_{\pi, \sigma \in \mathbb{S}_{q/2}} \prod_{k=1}^{q/4} A[(i_{\pi(2k-1)}, i_{\pi(2k)}), (j_{\sigma(2k-1)}, j_{\sigma(2k)})].
\end{aligned}$$

[2] The independent and concurrent work of [16] uses the same representation.

To formally show $\|M\|_2 = \tilde{O}(n/\sqrt{q})^{q/4}$ with high probability, we use the trace method to show

$$\mathbf{E}\left[\mathbf{Tr}(M^p)\right] \le 2^{O(pq\log p)} \frac{n^{pq/4+q/2}}{q^{pq/8}},$$

where $\mathbf{E}\left[\mathbf{Tr}(M^p)\right]$ can be written as (let $I^{p+1} := I^1$)

$$\begin{aligned}&\mathbf{E}\left[\sum_{I^1,\ldots,I^p\in[n]^{q/2}} \prod_{j=1}^{p} M[I^j, I^{j+1}]\right] \\ &= \sum_{I^1,\ldots,I^p} \mathbf{E}\left[\prod_{j=1}^{p}\Big(\sum_{\pi_j,\sigma_j\in\mathbb{S}_{q/2}} \prod_{k=1}^{q/4} A[(I^k_{\pi_j(2k-1)}, I^k_{\pi_j(2k)}), (I^{k+1}_{\sigma_j(2k-1)}, I^{k+1}_{\sigma_j(2k)})]\Big)\right].\end{aligned}$$

Let $E(I^1,\ldots,I^p)$ be the expectation value for $I^1,\ldots,I^p$ in the right hand side. We study $E(I^1,\ldots,I^p)$ for each $I^1,\ldots,I^p$ by careful counting of the number of permutations on a given sequence with possibly repeated entries. For any $I^1,\ldots,I^p \in [n]^{q/2}$, let $\#\left(I^1,\ldots,I^p\right)$ denote the number of distinct elements of $[n]$ that occur in $I^1,\ldots,I^p$, and for each $s = 1,\ldots,\#\left(I^1,\ldots,I^p\right)$, let $c^s \in (\{0\}\cup[q/2])^p$ denote the number of times that the jth smallest element occurs in $I^1,\ldots,I^p$. When $E(I^1,\ldots,I^p) \ne 0$, it means that for some permutations $\{\pi_j,\sigma_j\}_j$, every term $A[\cdot,\cdot]$ must appear even number of times. This implies that the number of distinct elements in $I^1,\ldots,I^p$ is at most half the maximal possible number $pq/2$. This lemma proves the intuition via graph theoretic arguments.

▶ **Lemma 9.** *If* $E(I^1,\ldots,I^p) \ne 0$, $\#\left(I^1,\ldots,I^p\right) \le \frac{pq}{4} + \frac{q}{2}$.

The number of $I^1,\ldots,I^p$ that corresponds to a sequence $c^1,\ldots,c^s$ is at most $\frac{n^s}{s!}\cdot\frac{((q/2)!)^p}{\prod_{\ell\in[p]} c^1_\ell!\cdot c^p_\ell!}$. Furthermore, there are at most $2^{O(pq)}p^{pq/2}$ different choices of $c^1,\ldots,c^s$ that corresponds to some $I^1,\ldots,I^p$. The following technical lemma bounds $E(I^1,\ldots,I^p)$ by careful counting arguments.

▶ **Lemma 10.** *For any* $I^1,\ldots,I^p$, $E(I^1,\ldots,I^p) \le 2^{O(pq)}\frac{p^{5pq/8}}{q^{3pq/8}}\prod_{\ell\in[p]} c^1_\ell!\ldots c^s_\ell!$.

Summing over all s and multiplying all possibilities,

$$\begin{aligned}\mathbf{E}\left[\mathbf{Tr}(M^p)\right] &\le \sum_{s=1}^{pq/4+q/2} \left(2^{O(pq)}p^{pq/2}\right)\cdot\left(\frac{n^s}{s!}\cdot((q/2)!)^p\right)\cdot\left(2^{O(pq)}\frac{p^{5pq/8}}{q^{3pq/8}}\right) \\ &= \max_{1\le s\le pq/4+q/2} 2^{O(pq\log p)}\cdot n^s\cdot\frac{q^{pq/8}}{s!}.\end{aligned}$$

When $q \le n$, the maximum occurs when $s = pq/4+q/2$, so $\mathbf{E}\left[\mathbf{Tr}(M^p)\right] \le 2^{O(pq\log p)}\cdot\frac{n^{pq/4+q/2}}{q^{pq/8}}$ as desired.

3.2 Overview of Lower Bound Proofs

Let $\mathcal{A}, A, f$ be as in Section 3.1. To prove the lower bound in Theorem 1, we construct a moment matrix M that is positive semidefinite, SoS-symmetric, $\mathbf{Tr}(\mathsf{M}) = 1$, and $\langle A, \mathsf{M}\rangle \ge 2^{-O(d)}\cdot\frac{n^{d/4}}{d^{d/4}}$. At a high level, our construction is $\mathsf{M} := c_1\mathsf{A} + c_2\mathsf{W}$ for some c_1, c_2, where A contains entries of A only corresponding to the multilinear indices, averaged over all SoS-symmetric positions. This gives a large inner product with A, SoS-symmetry, and nice spectral properties even though it is not positive semidefinite. The most natural way to make it positive semidefinite is adding a copy of the identity matrix, but this will again break the SoS-symmetry.

Our main technical contribution here is the construction of W that acts like a *SoS-symmetrized identity*. It has the minimum eigenvalue at least $\frac{1}{2}$, while the trace being $n^{d/2} \cdot 2^{O(d)}$, so the ratio of the average eigenvalue to the minimum eigenvalue is bounded above by $2^{O(d)}$, which allows us to prove a tight lower bound. To the best of our knowledge, no such bound was known for SoS-symmetric matrices except small values of $d = 3, 4$.

Given $I, J \in [n]^{d/2}$, we let $\mathsf{W}[I, J] := \mathbf{E}[x^{\alpha(I)+\alpha(J)}]$, where $x_1, \ldots, x_n$ are independently sampled from the *Wigner semicircle distribution*, whose probability density function is the semicircle $f(x) = \frac{2}{\pi}\sqrt{1 - x^2}$. Since $\mathbf{E}[x_1^{\ell}] = 0$ if ℓ is odd and $\mathbf{E}[x_1^{2\ell}] = \frac{1}{\ell+1}\binom{2\ell}{\ell}$, which is the ℓth Catalan number, each entry of W is bounded by $2^{O(d)}$ and $\mathbf{Tr}(\mathsf{W}) \leq n^{d/2} \cdot 2^{O(d)}$. To prove a lower bound on the minimum eigenvalue, we show that for any degree-ℓ polynomial p with m variables, $\mathbf{E}[p(x_1, \ldots, x_m)^2]$ is large by induction on ℓ and m. We use another property of the Wigner semicircle distribution that if $H \in \mathbb{R}^{(d+1)\times(d+1)}$ is the univariate moment matrix of x_1 defined by $H[i, j] = \mathbf{E}[x_1^{i+j}]$ $(0 \leq i, j \leq d)$ and $H = (R^T)R$ is the Cholesky decomposition of H, R is an upper triangular matrix with 1's on the main diagonal. This nice Cholesky decomposition allows us to perform the induction on the number of variables while the guarantee on the minimum eigenvalue is independent of n.

4 Upper bounds for even degree tensors

For even d, let $\mathcal{A} \in \mathbb{R}^{[n]^d}$ be a d-tensor with i.i.d. ±1 entries and let $A \in \mathbb{R}^{[n]^{d/2} \times [n]^{d/2}}$ be the matrix flattening of $\mathcal{A}$, i.e., $A[I, J] = \mathcal{A}[I \oplus J]$ (recall that $\oplus$ denotes tuple concatenation). Also let $f(x) := \mathcal{A}(x) = \langle \mathcal{A}, x^{\otimes d}\rangle$. With high probability $f_{\max} = O(\sqrt{n \cdot d \cdot \log d})$. In this section, we prove that for every q divisible by d, with high probability,

$$\left(\Lambda\left(f^{q/d}\right)\right)^{d/q} \;\leq\; \widetilde{O}\left(\frac{n}{q^{1-2/d}}\right)^{d/4} \;=\; \widetilde{O}\left(\frac{n}{q}\right)^{d/4-1/2} \cdot f_{\max}.$$

To prove it, we use the following matrix representation M of $f^{q/d}$, and show that $\|M\|_2 \leq \tilde{O}_d\left(\left(\frac{n\log^5 n}{q^{1-2/d}}\right)^{q/4}\right)$. Given a tuple $I = (i_1, \ldots, i_q)$, and an integer d that divides q and $1 \leq \ell \leq q/d$, let $I_{\ell;d}$ be the d-tuple $(I_{d(\ell-1)+1}, \ldots, I_{d\ell})$ (i.e., if we divide I into q/d tuples of length d, $I_{\ell;d}$ be the ℓ-th tuple). Furthermore, given a tuple $I = (i_1, \ldots, i_q) \in [n]^q$ and a permutation $\pi \in [n]^q$, let $\pi(I)$ be another q-tuple whose ℓth coordinate is $\pi(i_\ell)$. For $I, J \in [n]^{q/2}$, $M[I, J]$ is formally given by

$$\begin{aligned} M[I, J] &= \frac{1}{q!} \cdot \sum_{\pi,\sigma \in \mathbb{S}_{q/2}} A^{\otimes q/d}[\pi(I), \sigma(J)] \\ &= \frac{1}{q!} \cdot \sum_{\pi,\sigma \in \mathbb{S}_{q/2}} \prod_{\ell=1}^{q/d} A[(\pi(I))_{\ell;d/2}, (\sigma(J))_{\ell;d/2}]. \end{aligned}$$

We perform the trace method to bound $\|M\|_2$. Let p be an even integer, that will be eventually taken as $\Theta(\log n)$. $\mathbf{Tr}(M)$ can be written as (let $I^{p+1} := I^1$)

$$\mathbf{E}\left[\sum_{I^1,\ldots,I^p \in [n]^{q/2}} \prod_{\ell=1}^{p} M[I^\ell, I^{\ell+1}]\right]$$

$$= \sum_{I^1,\dots,I^p} \mathbf{E}\left[\prod_{\ell=1}^{p}(\sum_{\pi_j,\sigma_j\in\mathbb{S}_{q/2}} \prod_{m=1}^{q/d} A[(\pi(I^\ell))_{m;d/2},(\sigma(I^{\ell+1}))_{m;d/2})])\right].$$

Let $E(I^1,\dots,I^p) := \mathbf{E}\left[\prod_{\ell=1}^{p} M[I^\ell, I^{\ell+1}]\right]$, which is the expected value in the right hand side. To analyze $E(I^1,\dots,I^p)$, we first introduce notions to classify $I^1,\dots,I^p$ depending on their intersection patterns. For any $I^1,\dots,I^p \in [n]^{q/2}$, let e_k denote the k-th smallest element in $\bigcup\limits_{\ell,j}\{i_j^\ell\}$. For any $c^1,\dots,c^s \in [q/2]^p$, let

$$\mathcal{C}(c^1\dots c^s) :=$$
$$\left\{(I^1,\dots,I^p)\ \middle|\ \#\left(I^1,\dots,I^p\right) = s,\ \forall k\in[s], \ell\in[p],\ e_k \text{ appears } c_\ell^k \text{ times in } I^\ell\right\}.$$

The following two observations on $c^1,\dots,c^s$ can be easily proved.

▶ **Observation 11.** *If $\mathcal{C}(c^1,\dots,c^s) \neq \phi$,*

$$\left|\mathcal{C}(c^1,\dots,c^s)\right| \leq \frac{n^s}{s!} \times \frac{((q/2)!)^p}{\prod\limits_{\ell\in[p]} c_\ell^1!\dots c_\ell^s!}.$$

Moreover,

$$\left|\left\{(c^1,\dots,c^s)\in([q/2]^p)^s\ \middle|\ \mathcal{C}(c^1,\dots,c^s)\neq\phi\right\}\right| \leq 2^{O(pq)}p^{pq/2}.$$

The following lemma bounds $E(I^1,\dots,I^p)$ in terms of the corresponding $c_1,\dots,c_s$.

▶ **Lemma 12.** *Consider any $c^1,\dots,c^s \in [q/2]^p$ and $(I^1,\dots,I^p)\in\mathcal{C}(c^1,\dots,c^s)$. We have*

$$E(I^1,\dots,I^p) \leq 2^{O(pq)}\,\frac{p^{1/2+1/2d}}{q^{1/2-1/2d}} \prod_{\ell\in[p]} c_\ell^1!\dots c_\ell^s!$$

Proof. Consider any $c^1,\dots,c^s \in [q/2]^p$ and $(I^1,\dots,I^p)\in\mathcal{C}(c^1,\dots,c^s)$. We have

$$\begin{aligned}
&E(I^1,\dots,I^p)\\
&= \mathbf{E}\left[\prod_{\ell=1}^{p} M[I^\ell, I^{\ell+1}]\right]\\
&= \sum_{\pi_j,\sigma_j\in\mathbb{S}_{q/2}} \mathbf{E}\left[\prod_{\ell=1}^{p}\prod_{m=1}^{q/d} A[(\pi(I^\ell))_{m;d/2},(\pi(I^{\ell+1}))_{m;d/2}]\right]\\
&= \left(\frac{\prod_\ell\prod_s (c_\ell^s!)^2}{((q/2)!)^{2p}}\right)\cdot \sum_{(J^\ell,K^\ell\in\mathcal{O}(I^\ell))_{\ell\in[p]}} \mathbf{E}\left[\prod_{\ell=1}^{p}\prod_{m=1}^{q/d} A[J^\ell_{m;d/2}, K^{\ell+1}_{m;d/2}]\right] \qquad (4.1)
\end{aligned}$$

Thus, $E(I^1,\dots,I^p)$ is bounded by the number of choices for $J^1,\dots,J^p,K^1,\dots,K^p$ such that $J^\ell, K^\ell \in \mathcal{O}\left(I^\ell\right)$ for each $\ell\in[p]$, and $\mathbf{E}\left[\prod_{\ell=1}^{p}\prod_{m=1}^{q/d} A[J^\ell_{m;d/2}, K^{\ell+1}_{m;d/2}]\right]$ is nonzero.

Given $J^1,\dots,J^p$ and $K^1,\dots,K^p$, consider the (pq/d)-tuple T where each coordinate is indexed by $(\ell,m)_{\ell\in[p],m\in[q/d]}$ and has a d-tuple $T_{\ell,m} := (J^\ell_{m;d/2}) \oplus (K^{\ell+1}_{m;d/2}) \in \mathbb{R}^d$ as a value. Note that $\sum_{\ell,m}\alpha(T_{\ell,m})) = (2o_1,\dots,2o_n)$ where o_r is the number of occurences of $r\in[n]$ in $(pq/2)$-tuple $\oplus_{\ell=1}^{p} I^\ell$. The fact that $\mathbf{E}\left[\prod_{\ell=1}^{p}\prod_{m=1}^{q/d} A[j_{m;d/2}, k_{m;d/2}]\right] \neq 0$ means that every d-tuple occurs even number of times in T.

We count the number of (pq/d)-tuples $T = (T_{\ell,m})_{\ell\in[p],m\in[q]}$ that $\sum_{\ell,m}\alpha(T_{\ell,m}) = (2o_1,\dots,2o_n)$ and every d-tuple occurs an even number of times. Let $Q = (Q_1,\dots,Q_{pq/2d})$, $R = (R_1,\dots,R_{pq/2d})$ be two $(pq/2d)$-tuples of d-tuples where for every d-tuple P, the number of occurences of P is the same in Q and R, and $\sum_{\ell=1}^{pq/2d}\alpha(Q_\ell) = \sum_{\ell=1}^{pq/2d}\alpha(R_\ell) = (o_1,\dots,o_n)$. At most $2^{pq/d}$ tuples T can be made by *interleaving* Q and R – for each (ℓ,m), choose $T_{\ell,m}$ from the first unused d-tuple in either Q or R. Furthermore, every tuple T that meets our condition can be constructed in this way.

Due to the condition $\sum_{\ell=1}^{pq/2d}\alpha(Q_\ell) = (o_1,\dots,o_n)$, the number of choices for Q is at most the number of different ways to permute $I^1\oplus\cdots\oplus I^p$, which is at most $(pq/2)!/\prod_{m\in[s]}(\bar{c}^m)!$, where $\bar{c}^m := \sum_{\ell\in[p]} c_\ell^m$ for $m\in[s]$. For a fixed choice of Q, there are at most $(pq/2d)!$ choices of R. Therefore, the number of choices for $(J^\ell, K^\ell \in \mathcal{O}(I^\ell))_{\ell\in[p]}$ with nonzero expected value is at most

$$2^{pq/d}\cdot\frac{(pq/2)!}{\prod_{m\in[s]}(\bar{c}^m)!}\cdot(pq/2d)! = 2^{O(pq)}\cdot\frac{(pq)^{1/2+1/2d}}{\prod_{m\in[s]}(\bar{c}^m)!}.$$

Combining with Eq. (4.1),

$$E(I^1,\dots,I^p) \le \left(2^{O(pq)}\frac{(pq)^{1/2+1/2d}}{\prod_{m\in[s]}(\bar{c}^m)!}\right)\cdot\left(\frac{\prod_\ell\prod_s(c_\ell^s!)^2}{((q/2)!)^{2p}}\right) \le 2^{O(pq)}\cdot\frac{p^{1/2+1/2d}}{q^{1/2-1/2d}}\cdot\prod_\ell\prod_s c_\ell^s!$$

as desired. ◀

▶ **Lemma 13.** *For all $I^1,\dots,I^p\in[n]^{q/2}$, if $E(I^1,\dots,I^p)\neq 0$, $\#\left(I^1,\dots,I^p\right)\le\frac{pq}{4}+\frac{q}{2}$.*

Proof. Note that $E(I^1,\dots,I^p)\neq 0$ implies that there exist $J^1,\dots,J^p,K^1,\dots,K^p$ such that $J^\ell,K^\ell\in\mathcal{O}(I^\ell)$ and every d-tuple occurs exactly even number of times in $((J^\ell_{m;d/2})\oplus(K^{\ell+1}_{m;d/2}))_{\ell\in[p],m\in[q/d]}$. Consider the graph $G=(V,E)$ defined by

$$\begin{aligned} V &:= \bigcup_{\ell\in[p]}\bigcup_{k\in[q/2]}\{I_k^\ell\}\\ E &:= \bigcup_{m\in[q/2]}\left\{\{J_m^1,K_m^2\},\{J_m^2,K_m^3\},\dots,\{J_m^p,K_m^1\}\right\}. \end{aligned}$$

The even multiplicity condition implies that every element in E has even multiplicity and consequently $|E|\le pq/4$. We next show that E is the union of $q/2$ paths. To this end, we construct $G^1\in\mathcal{O}(I^1),\dots,G^\ell\in\mathcal{O}(I^\ell)$ as follows:

1. Let $G^2 := K^2$
2. For $3\le\ell\le p$ do:
   a. Since $G^\ell\in\mathcal{O}(J^\ell)$, there exists $\pi\in\mathbb{S}_{q/2}$ s.t. $\pi(J^\ell)=G^\ell$.
   b. Let $G^{\ell+1} := \pi(K^{\ell+1})$.

We observe that by construction,

$$\begin{aligned} &\bigcup_{m\in[q/2]}\left\{\{J_m^1,G_m^2\},\{G_m^2,G_m^3\},\dots,\{G_m^p,G_m^1\}\right\}\\ =&\bigcup_{m\in[q/2]}\left\{\{J_m^1,K_m^2\},\{J_m^2,K_m^3\},\dots,\{J_m^p,K_m^1\}\right\} = E \end{aligned}$$

which establishes that E is a union of $q/2$ paths.

Now since E is the union of $q/2$ paths G has at most $q/2$ connected components, and one needs to add at most $q/2-1$ edges make it connected, we have $|V|\le|E|+(q/2-1)+1\le pq/4+q/2$. But $\#\left(I^1,\dots,I^p\right)=|V|$, which completes the proof. ◀

Finally, $\mathbf{E}\left[\mathbf{Tr}(M^p)\right]$ can be bounded as follows.

$$\begin{aligned}
&\mathbf{E}\left[\mathbf{Tr}(M^p)\right] \\
&= \sum_{I^1,\dots,I^p \in [n]^{q/2}} E(I^1,\dots,I^p) \\
&= \sum_{s\in[pq/4+q/2]} \sum_{\#(I^1,\dots,I^p)=s} E(I^1,\dots,I^p) && \text{(by Lemma 13)} \\
&= \sum_{s\in[pq/4+q/2]} \sum_{c^1,\dots,c^s\in[q/2]^p} \sum_{(I^1,\dots,I^p)\in\mathcal{C}(c^1\dots c^s)} E(I^1,\dots,I^p) \\
&= \sum_{s\in[pq/4+q/2]} \sum_{c^1,\dots,c^s\in[q/2]^p} \sum_{(I^1,\dots,I^p)\in\mathcal{C}(c^1\dots c^s)} E(I^1,\dots,I^p) \\
&\le \sum_{s\in[pq/4+q/2]} \sum_{c^1,\dots,c^s\in[q/2]^p} \\
&\quad \sum_{(I^1,\dots,I^p)\in\mathcal{C}(c^1\dots c^s)} 2^{O(pq)} \frac{p^{(1/2+1/2d)pq}}{q^{(1/2-1/2d)pq}} \prod_{\ell\in[p]} c^1_\ell! \dots c^s_\ell! && \text{(by Lemma 12)} \\
&\le \sum_{s\in[pq/4+q/2]} 2^{O(pq)} \frac{n^s}{s!} p^{(1+1/2d)pq} q^{pq/2d} && \text{(by Observation 11)} \\
&\le \sum_{s\in[pq/4+q/2]} 2^{O(pq)} \frac{n^{pq/4+q/2}}{s!\, q^{pq/4+q/2-s}} p^{(1/2+1/2d)p1} q^{(1/2-1/2d)pq} && \text{(assuming } q \le n\text{)} \\
&\le \sum_{s\in[pq/4+q/2]} 2^{O(pq)} \frac{n^{pq/4+q/2}\, p^{(1+1/2d)pq}}{q^{(1/4-1/2d)pq}} \\
&\le 2^{O(pq)} \frac{n^{pq/4+q/2}\, p^{(1+1/2d)pq}}{q^{(1/4-1/2d)pq}}.
\end{aligned}$$

Choose p to be even and let $p = \Theta(\log n)$. Applying Markov inequality shows that with high probability,

$$\left(\Lambda\left(f^{q/d}\right)\right)^{d/q} \le \left(\|M\|_2\right)^{d/q} \le \left(\mathbf{E}\left[\mathbf{Tr}(M^p)\right]\right)^{d/pq} = O_d\left(\frac{n^{d/4}\cdot(\log n)^{\,d+1/2}}{q^{\,d/4-1/2}}\right).$$

Thus we obtain

▶ **Theorem 14.** *For even d, let $\mathcal{A} \in \mathbb{R}^{[n]^d}$ be a d-tensor with i.i.d. ± 1 entries. Then for any even q such that $q \le n$, we have that with probability $1 - n^{\Omega(1)}$,*

$$\frac{\mathsf{SoS}_q(\mathcal{A}(x))}{\mathcal{A}_{\max}} \le \left(\frac{\widetilde{O}(n)}{q}\right)^{d/4-1/2}.$$

5 Proof of SoS Lower Bound in Theorem 1

For even q, let $\mathcal{A} \in \mathbb{R}^{[n]^q}$ be a q-tensor with i.i.d. ± 1 entries and let $A \in \mathbb{R}^{[n]^{q/2}\times[n]^{q/2}}$ be the matrix flattening of $\mathcal{A}$, i.e., $A[I,J] = \mathcal{A}[I \oplus J]$ (recall that $\oplus$ denotes tuple concatenation). Also let $f(x) := \mathcal{A}(x) = \langle \mathcal{A}, x^{\otimes q}\rangle$. This section proves the lower bound in Theorem 1, by constructing a moment matrix M that is positive semidefinite, SoS-symmetric, $\mathbf{Tr}(\mathsf{M}) = 1$, and $\langle A, \mathsf{M}\rangle \ge 2^{-O(q)} \cdot \frac{n^{q/4}}{q^{q/4}}$. In Section 5.1, we construct the matrix $\widehat{\mathsf{W}}$ that acts as a SoS-symmetrized identity matrix. The moment matrix M is presented in Section A.

5.1 Wigner Moment Matrix

In this section, we construct an SoS-symmetric and positive semidefinite matrix $\widehat{\mathsf{W}} \in \mathbb{R}^{\mathbb{N}^n_{q/2} \times \mathbb{N}^n_{q/2}}$ such that $\lambda_{\min}(\widehat{\mathsf{W}}) / \mathbf{Tr}\left(\widehat{\mathsf{W}}\right) \geq 1/(2^{q+1} \cdot |\mathbb{N}^n_{q/2}|)$, i.e. the ratio of the minimum eigenvalue to the average eigenvalue is at least $1/2^{q+1}$.

▶ **Theorem 15.** *For any positive integer n and any positive even integer q, there exists a matrix $\widehat{\mathsf{W}} \subseteq \mathbb{R}^{\mathbb{N}^n_{q/2} \times \mathbb{N}^n_{q/2}}$ that satisfies the following three properties: (1) $\widehat{\mathsf{W}}$ is degree-q SoS symmetric. (2) The minimum eigenvalue of $\widehat{\mathsf{W}}$ is at least $\frac{1}{2}$. (3) Each entry of $\widehat{\mathsf{W}}$ is in $[0, 2^q]$.*

Theorem 15 is proved by explicitly constructing independent random variables $x_1, \ldots, x_n$ such that for any n-variate polynomial $p(x_1, \ldots, x_n)$ of degree at most $\frac{q}{2}$, $\mathbf{E}[p^2]$ is bounded away from 0. The proof consists of three parts. The first part shows the existence of a desired distribution for one variable x_i. The second part uses induction to prove that $\mathbf{E}[p^2]$ is bounded away from 0. The third part constructs $\widehat{\mathsf{W}} \subseteq \mathbb{R}^{\mathbb{N}^n_{q/2} \times \mathbb{N}^n_{q/2}}$ from the distribution defined.

Wigner Semicircle Distribution and Hankel Matrix

Let k be a positive integer. In this part, the rows and columns of all $(k+1) \times (k+1)$ matrices are indexed by $\{0, 1, \ldots, k\}$. Let T be a $(k+1) \times (k+1)$ matrix where $T[i,j] = 1$ if $|i-j| = 1$ and $T[i,j] = 0$ otherwise. Let $e_0 \in \mathbb{R}^{k+1}$ be such that $(e_0)_0 = 1$ and $(e_0)_i = 0$ for $1 \leq i \leq k$. Let $R \in \mathbb{R}^{(k+1)\times(k+1)}$ be defined by $R := [e_0, Te_0, T^2 e_0, \ldots, T^k e_0]$. Let $R_0, \ldots, R_k$ be the columns or R so that $R_i = T^i e_0$. It turns out that R is closely related to the number of ways to consistently put parantheses. Given a string of parantheses '(' or ')', we call it *consistent* if any prefix has at least as many '(' as ')'. For example, ((())(is consistent, but ())((is not.

▶ **Claim 16.** *$R[i,j]$ is the number of ways to place j parantheses '(' or ')' consistently so that there are i more '(' than ')'.*

Proof. We proceed by the induction on j. When $j = 0$, $R[0,0] = 1$ and $R[i,0] = 0$ for all $i \geq 1$. Assume the claim holds up to $j-1$. By the definition $R_j = TR_{j-1}$.

- For $i = 0$, the last parenthesis must be the close parenthesis, so the definition $R[0,j] = R[1,j-1]$ still measures the number of ways to place j parantheses with equal number of '(' and ')'.
- For $i = k$, the last parenthesis must be the open parenthesis, so the definition $R[k,j] = R[k-1,j-1]$ still measures the number of ways to place j parantheses with k more '('.
- For $0 < i < k$, the definition of R gives $R[i,j] = R[i-1,j-1] + R[i+1,j-1]$. Since $R[i-1,j]$ corresponds to plaincg ')' in the jth position and $R[i+1,j]$ corresponds to placing '(' in the jth position, $R[i,j]$ still measures the desired quantity.

This completes the induction and proves the claim. ◀

Easy consequences of the above claim are (1) $R[i,i] = 1$ for all $0 \leq i \leq k$, and $R[i,j] = 0$ for $i > j$, and (2) $R[i,j] = 0$ if $i+j$ is odd, and $R[i,j] \geq 1$ if $i \leq j$ and $i+j$ is even.

Let $H := (R^T)R$. Since R is upper triangular with 1's on the main diagonal, $H = (R^T)R$ gives the unique Cholesky decomposition, so H is positive definite. It is easy to see that $H[i,j] = \langle R_i, R_j \rangle$ is the total number of ways to place $i+j$ parantheses consistently with the same number of '(' and ')'. Therefore, $H[i,j] = 0$ if $i+j$ is odd, and if $i+j$ is even (let $l := \frac{i+j}{2}$), $H[i,j]$ is the lth Catalan number $C_l := \frac{1}{l+1}\binom{2l}{l}$. In particular, $H[i,j] = H[i',j']$ for all $i+j = i'+j'$. Such H is called a *Hankel matrix.*

Given a sequence of $m_0 = 1, m_1, m_2, \ldots$ of real numbers, the *Hamburger moment problem* asks whether there exists a random variable W supported on $\mathbb{R}$ such that $\mathbf{E}[W^i] = m_i$. It is well-known that there exists a unique such W if for all $k \in \mathbb{N}$, the Hankel matrix $H \in \mathbb{R}^{(k+1)\times(k+1)}$ defined by $H[i,j] := \mathbf{E}[W^{i+j}]$ is positive definite [17]. Since our construction of $H \in \mathbb{R}^{(k+1)\times(k+1)}$ ensures its positive definiteness for any $k \in \mathbb{N}$, there exists a unique random variable W such that $\mathbf{E}[W^i] = 0$ if i is odd, $\mathbf{E}[W^i] = C_{\frac{i}{2}}$ if i is even. It is known as the *Wigner semicircle distribution* with radius $R = 2$.

▶ Remark. Some other distributions (e.g., Gaussian) will give an asymptotically weaker bound. Let G be a standard Gaussian random variable. The quantitative difference comes from the fact that $\mathbf{E}[W^{2l}] = C_l = \frac{1}{l+1}\binom{2l}{l} \leq 2^l$ while $\mathbf{E}[G^{2l}] = (2l-1)!! \geq 2^{\Omega(l \log l)}$.

Multivariate Distribution

Fix n and q. Let $k = \frac{q}{2}$. Let $H \in \mathbb{R}^{(k+1)\times(k+1)}$ be the Hankel matrix defined as above, and W be a random variable sampled from the Wigner semicircle distribution. Consider $x_1, \dots, x_n$ where each x_i is an independent copy of $\frac{W}{N}$ for some large number N to be determined later. Our $\widehat{\mathsf{W}}$ is later defined to be $\widehat{\mathsf{W}}[\alpha, \beta] = \mathbf{E}[x^{\alpha+\beta}] \cdot N^q$ so that the effect of the normalization by N is eventually cancelled, but large N is needed to prove the induction that involves non-homogeneous polynomials.

We study $\mathbf{E}[p(x)^2]$ for any n-variate (possibly non-homogeneous) polynomial p of degree at most k. For a multivarite polynomial $p = \sum_{\alpha \in \mathbb{N}^n_{\leq k}} p_\alpha x^\alpha$, define ℓ_2 norm of p to be $\|p\|_{\ell_2} := \sqrt{\sum_\alpha p_\alpha^2}$. For $0 \leq m \leq n$ and $0 \leq l \leq k$, let $\sigma(m,l) := \inf_p \mathbf{E}[p(x)^2]$ where the infimum is taken over polynomials p such that $\|p\|_{\ell_2} = 1$, $\deg(p) \leq l$, and p depends only on $x_1, \dots, x_m$.

▶ **Lemma 17.** *There exists $N := N(n,k)$ such that $\sigma(m,l) \geq \frac{(1-\frac{m}{2n})}{N^{2l}}$ for all $0 \leq m \leq n$ and $0 \leq l \leq k$.*

Proof. We prove the lemma by induction on m and l. When $m = 0$ or $l = 0$, p becomes the constant polynomial 1 or -1, so $\mathbf{E}[p^2] = 1$.

Fix $m, l > 0$ and a polynomial $p = p(x_1, \dots, x_m)$ of degree at most l. Decompose $p = \sum_{i=0}^{l} p_i x_m^i$ where each p_i does not depend on x_m. The degree of p_i is at most $l - i$.

$$\mathbf{E}[p^2] = \mathbf{E}[(\sum_{i=0}^{l} p_i x_m^i)^2] = \sum_{0 \leq i,j \leq l} \mathbf{E}[p_i p_j]\, \mathbf{E}[x_m^{i+j}].$$

Let $\Sigma = \mathsf{diag}(1, \frac{1}{N}, \dots, \frac{1}{N^l}) \in \mathbb{R}^{(l+1)\times(l+1)}$. Let $H_l \in \mathbb{R}^{(l+1)\times(l+1)}$ be the submatrix of H with the first $l+1$ rows and columns. The rows and columns of $(l+1)\times(l+1)$ matrices are still indexed by $\{0, \dots, l\}$. Define $R_l \in \mathbb{R}^{(l+1)\times(l+1)}$ similarly from R, and r_t $(0 \leq t \leq l)$ be the tth column of $(R_l)^T$. Note $H_l = (R_l)^T R_l = \sum_{t=0}^{l} r_t r_t^T$. Let $H' = \Sigma H_l \Sigma$ such that $H'[i,j] = \mathbf{E}[x_m^{i+j}]$. Finally, let $P \in \mathbb{R}^{(l+1)\times(l+1)}$ be defined such that $P[i,j] := \mathbf{E}[p_i p_j]$. Then $\mathbf{E}[p^2]$ is equal to

$$\mathbf{Tr}(PH') = \mathbf{Tr}(P\Sigma H_l \Sigma) = \mathbf{Tr}\left(P\Sigma(\sum_{t=0}^{l} r_t r_t^T)\Sigma\right)$$
$$= \sum_{t=0}^{l} \mathbf{E}[(p_t \frac{1}{N^t} + p_{t+1}\frac{(r_t)_{t+1}}{N^{t+1}} + \cdots + p_l \frac{(r_t)_l}{N^l})^2],$$

where the last step follows from the fact that $(r_t)_j = 0$ if $j < t$ and $(r_t)_t = 1$. Consider the polynomial

$$q_t := p_t \frac{1}{N^t} + p_{t+1}\frac{(r_t)_{t+1}}{N^{t+1}} + \cdots + p_l \frac{(r_t)_l}{N^l}.$$

Since p_i is of degree at most $l-i$, q_t is of degree at most $l-t$. Also recall that each entry of R is bounded by 2^k. By the triangle inequality,

$$\|q_t\|_{\ell_2} \geq \frac{1}{N^t}\left(\|p_t\|_{\ell_2} - \big(\|p_{t+1}\|_{\ell_2} \frac{(r_t)_{t+1}}{N} + \cdots + \|p_l\|_{\ell_2} \frac{(r_t)_l}{N^{l-t}} \big) \right) \geq \frac{1}{N^t}\left(\|p_t\|_{\ell_2} - \frac{k2^k}{N} \right),$$

and

$$\|q_t\|_{\ell_2}^2 \geq \frac{1}{N^{2t}}\left(\|p_t\|_{\ell_2}^2 - \frac{2k2^k}{N} \right).$$

Finally,

$$\begin{aligned}
\mathbf{E}[p^2] &= \sum_{t=0}^{l} \mathbf{E}[q_t^2] \\
&\geq \sum_{t=0}^{l} \sigma(m-1, l-t) \cdot \|q_t\|_{\ell_2}^2 \\
&\geq \sum_{t=0}^{l} \sigma(m-1, l-t) \cdot \frac{1}{N^{2t}} \left(\|p_t\|_{\ell_2}^2 - \frac{2k2^k}{N} \right) \\
&\geq \sum_{t=0}^{l} \frac{(1-\frac{m-1}{2n})}{N^{2l-2t}} \cdot \frac{1}{N^{2t}} \cdot \left(\|p_t\|_{\ell_2}^2 - \frac{2k2^k}{N} \right) \\
&= \frac{(1-\frac{m-1}{2n})}{N^{2l}} \cdot \sum_{t=0}^{l} \left(\|p_t\|_{\ell_2}^2 - \frac{2k2^k}{N} \right) \\
&\geq \frac{(1-\frac{m-1}{2n})}{N^{2l}} \cdot (1 - \frac{2K^2 2^k}{N}).
\end{aligned}$$

Take $N := 4nK^2 2^k$ so that $\big(1-\frac{m-1}{2n}\big) \cdot \big(1 - \frac{2K^2 2^k}{N}\big) \geq 1 - \frac{m-1}{2n} - \frac{2K^2 2^k}{N} = 1 - \frac{m}{2n}$. This completes the induction and proves the lemma. ◀

Construction of $\widehat{\mathsf{W}}$

We now prove Theorem 15. Given n and q, let $k = \frac{q}{2}$, and consider random variables $x_1, \ldots, x_n$ above. Let $\widehat{\mathsf{W}} \in \mathbb{R}^{\mathbb{N}_k^n \times \mathbb{N}_k^n}$ be such that for any $\alpha, \beta \in \mathbb{N}_k^n$, $\widehat{\mathsf{W}}[\alpha, \beta] = \mathbf{E}[x^{\alpha+\beta}] \cdot N^{2k}$. By definition, $\widehat{\mathsf{W}}$ is degree-q SoS symmetric. Since each entry of $\widehat{\mathsf{W}}$ corresponds to a monomial of degree exactly q and each x_i is drawn independently from the Wigner semicircle distribution, each entry of $\widehat{\mathsf{W}}$ is at most the $\frac{q}{2}$th Catalan number $C_{\frac{q}{2}} \leq 2^q$. For any unit vector $p = (p_S)_{S \in \mathbb{N}_k^n} \in \mathbb{R}^{\mathbb{N}_k^n}$, Lemma 17 shows $p^T \widehat{\mathsf{W}} p = \mathbf{E}[p^2] \cdot N^{2k} \geq \frac{1}{2}$ where p also represents a degree-k homogeneous polynomial $p(x_1, \ldots, x_n) = \sum_{\alpha \in \binom{[n]}{k}} p_\alpha x^\alpha$. Therefore, the minimum eigenvalue of $\widehat{\mathsf{W}}$ is at least $\frac{1}{2}$.

Due to space constraints, we defer the final construction of the moment matrix to the appendix (see Section A).

References

1 Boaz Barak, Fernando G. S. L. Brandao, Aram W. Harrow, Jonathan Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 307–326. ACM, 2012.

2 Boaz Barak, Jonathan A. Kelner, and David Steurer. Rounding sum-of-squares relaxations. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 31–40. ACM, 2014.

3 Boaz Barak, Jonathan A. Kelner, and David Steurer. Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 143–151. ACM, 2015.

4 Boaz Barak and David Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. *arXiv preprint arXiv:1404.5236*, 2014.

5 Vijay Bhattiprolu, Mrinalkanti Ghosh, Venkatesan Guruswami, Euiwoong Lee, and Madhur Tulsiani. Weak decoupling, polynomial folds, and approximate optimization over the sphere. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:185, 2016. URL: https://eccc.weizmann.ac.il/report/2016/185/.

6 Vijay Bhattiprolu, Venkatesan Guruswami, and Euiwoong Lee. Certifying random polynomials over the unit sphere via sum of squares hierarchy. *arXiv preprint arXiv:1605.00903*, 2016.

7 Fernando G. S. L. Brandao and Aram W. Harrow. Quantum de finetti theorems under local measurements with applications. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 861–870. ACM, 2013.

8 S. Charles Brubaker and Santosh S. Vempala. Random tensors and planted cliques. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 406–419. Springer, 2009.

9 Alan Frieze and Ravi Kannan. A new approach to the planted clique problem. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 2. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2008.

10 Rong Ge and Tengyu Ma. Decomposing overcomplete 3rd order tensors using sum-of-squares algorithms. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, page 829, 2015.

11 Samuel B. Hopkins, Pravesh K. Kothari, Aaron Potechin, Prasad Raghavendra, Tselil Schramm, and David Steurer. Personal communication, 2017.

12 Samuel B. Hopkins, Jonathan Shi, and David Steurer. Tensor principal component analysis via sum-of-square proofs. In *Proceedings of The 28th Conference on Learning Theory*, pages 956–1006, 2015.

13 Pravesh K. Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. In *Proceedings of the 49th ACM Symposium on Theory of Computing*, 2017. To appear.

14 Monique Laurent. Sums of squares, moment matrices and optimization over polynomials. In *Emerging applications of algebraic geometry*, pages 157–270. Springer, 2009.

15 Andrea Montanari and Emile Richard. A statistical model for tensor PCA. In *Advances in Neural Information Processing Systems*, pages 2897–2905, 2014.

16 Prasad Raghavendra, Satish Rao, and Tselil Schramm. Strongly refuting random CSPs below the spectral threshold. In *Proceedings of the 49th ACM Symposium on Theory of Computing*, 2017. To appear.

17 Barry Simon. The classical moment problem as a self-adjoint finite difference operator. *Advances in Mathematics*, 137(1):82–203, 1998.

18 R. Tomioka and T. Suzuki. Spectral norm of random tensors. *ArXiv e-prints*, July 2014. arXiv:1407.1870.

A Constructing the Moment Matrix Realizing the Lower Bound

For even d, let $\mathcal{A} \in \mathbb{R}^{[n]^q}$ be a q-tensor with i.i.d. ± 1 entries and let $A \in \mathbb{R}^{[n]^{q/2} \times [n]^{q/2}}$ be the matrix flattening of $\mathcal{A}$, i.e., $A[I, J] = \mathcal{A}[I \oplus J]$ (recall that $\oplus$ denotes tuple concatenation). Also let $f(x) := \mathcal{A}(x) = \langle \mathcal{A}, x^{\otimes q} \rangle$. Our lower bound on $f_{\max}$ by is proved by constructing a moment matrix $\mathsf{M} \in \mathbb{R}^{[n]^{q/2} \times [n]^{q/2}}$ that satisfies

- $\mathbf{Tr}(\mathsf{M}) = 1$.
- $\mathsf{M} \succeq 0$.
- M is SoS-symmetric.
- $\langle A, \mathsf{M}\rangle \;\geq\; 2^{-O(q)} \cdot n^{q/4}/q^{q/4}$,

where $A \in \mathbb{R}^{[n]^{q/2}\times[n]^{q/2}}$ is any matrix representation of f (SoS-symmetry of M ensures $\langle A, \mathsf{M}\rangle$ does not depend on the choice of A).

Let A be the SoS-symmetric matrix such that for any $I = (i_1, \ldots, i_{q/2})$ and $J = (j_1, \ldots, j_{q/2})$,

$$\mathsf{A}[I,J] = \begin{cases} \frac{f_{\alpha(I)+\alpha(J)}}{q!}, & \text{if } i_1, \ldots, i_{q/2}, j_1, \ldots, j_{q/2} \text{ are all distinct.} \\ 0 & \text{otherwise.} \end{cases}$$

We bound $\|\mathsf{A}\|_2$ in two steps. Let $\widehat{\mathsf{A}}_Q \in \mathbb{R}^{\mathbb{N}^n_{q/2}\times\mathbb{N}^n_{q/2}}$ be the *quotient matrix* of A defined by

$$\widehat{\mathsf{A}}_Q[\beta,\gamma] := \mathsf{A}[I,J] \cdot \sqrt{|\mathcal{O}(\beta)| \cdot |\mathcal{O}(\gamma)|},$$

where $I, J \in [n]^{q/2}$ are such that $\beta = \alpha(I), \gamma = \alpha(J)$.

▶ **Lemma 18.** *With high probability,* $\|\widehat{\mathsf{A}}_Q\|_2 \leq\; 2^{O(q)} \cdot \frac{n^{q/4}}{q^{q/4}}$.

Proof. Consider any $y \in \mathbb{R}^{\mathbb{N}^n_{q/2}}$ s.t. $\|y\| = 1$. Since

$$\begin{aligned} y^T \cdot \widehat{\mathsf{A}}_Q \cdot y &= \sum_{\beta+\gamma\leq \mathbf{1}} \widehat{A}_Q[\beta,\gamma] \cdot y_\beta \cdot y_\gamma \\ &= \sum_{\beta+\gamma\leq \mathbf{1}} y_\beta \cdot y_\gamma \sum_{\substack{\alpha(I)+\alpha(J)\\=\beta+\gamma}} A[I,J] \cdot \frac{\sqrt{|\mathcal{O}(\beta)||\mathcal{O}(\gamma)|}}{|\mathcal{O}(\beta+\gamma)|} \\ &= \sum_{I,J\in[n]^{q/2}} A[I,J] \sum_{\substack{\beta+\gamma\leq\mathbf{1}\\ \beta+\gamma=\\ \alpha(I)+\alpha(J)}} \frac{\sqrt{|\mathcal{O}(\beta)||\mathcal{O}(\gamma)|}}{|\mathcal{O}(\beta+\gamma)|} \cdot y_\beta \cdot y_\gamma \end{aligned}$$

So $y^T \cdot \widehat{\mathsf{A}}_Q \cdot y$ is a sum of independent random variables

$$\sum_{I,J\in[n]^q} A[I,J] \cdot c_{I,J}$$

where each $A[I,J]$ is independently sampled from the Rademacher distribution and

$$c_{I,J} := \sum_{\substack{\beta+\gamma\leq\mathbf{1}\\ \beta+\gamma=\\ \alpha(I)+\alpha(J)}} \frac{\sqrt{|\mathcal{O}(\beta)||\mathcal{O}(\gamma)|}}{|\mathcal{O}(\beta+\gamma)|} \cdot y_\beta \cdot y_\gamma \,.$$

Fix any $I, J \in [n]^{q/2}$ and let $\alpha := \alpha(I) + \alpha(J)$. By Cauchy-Schwarz,

$$c^2_{I,J} \;\leq\; \left(\sum_{\beta+\gamma=\alpha} \frac{|\mathcal{O}(\beta)||\mathcal{O}(\gamma)|}{|\mathcal{O}(\alpha)|^2}\right) \cdot \Big(\sum_{\beta+\gamma=\alpha} y^2_\beta \cdot y^2_\gamma\Big) \;\leq\; \frac{2^{O(q)}}{|\mathcal{O}(\alpha)|} \cdot \sum_{\beta+\gamma=\alpha} y^2_\beta \cdot y^2_\gamma \;=:\; c^2_\alpha\,, \tag{A.1}$$

since there are at most $2^{O(q)}$ choices of β and γ with $\beta+\gamma=\alpha$, and $|\mathcal{O}(\beta)|\cdot|\mathcal{O}(\gamma)|\le|\mathcal{O}(\alpha)|$. Therefore, $y^T\cdot\widehat{\mathsf{A}}_Q\cdot y$ is the sum of independent random variables that are centred and always lie in the interval $[-1,+1]$. Furthermore, by Eq. (A.1), the total variance is

$$\sum_{I,J\in[n]^{q/2}} c_{I,J}^2 \;\le\; \sum_{\alpha\in\mathbb{N}^n_q} c_\alpha^2\cdot|\mathcal{O}(\alpha)| \;\le\; 2^{O(q)}\cdot\sum_{\beta,\gamma\in\mathbb{N}^n_{q/2}} y_\beta^2\cdot y_\gamma^2 \;=\; 2^{O(q)}\cdot\Big(\sum_{\beta\in\mathbb{N}^n_{q/2}} y_\beta^2\Big)^2 \;=\; 2^{O(q)}$$

The claim then follows from combining standard concentration bounds with a union bound over a sufficiently fine net of the unit sphere in $|\mathbb{N}^n_{q/2}|\le 2^{O(q)}\cdot\frac{n^{q/2}}{q^{q/2}}$ dimensions. ◀

▶ **Lemma 19.** *For any SoS-symmetric* $\mathsf{A}\in\mathbb{R}^{[n]^{q/2}\times[n]^{q/2}}$, $\|\mathsf{A}\|_2\le\left\|\widehat{\mathsf{A}}_Q\right\|_2$.

Proof. For any $u,v\in\mathbb{R}^{[n]^{q/2}}$ *s.t.* $\|u\|=\|v\|=1$, we have

$$\begin{aligned}
&u^T\mathsf{A}v\\
&=\sum_{I,J\in[n]^{q/2}}\mathsf{A}[I,J]u_Iv_J\\
&=\sum_{I,J\in[n]^{q/2}}\frac{\widehat{\mathsf{A}}_Q[\alpha(I),\alpha(J)]}{\sqrt{|\mathcal{O}(I)|\,|\mathcal{O}(J)|}}\cdot u_Iv_J\\
&=\sum_{\alpha,\beta\in\mathbb{N}^n_{q/2}}\frac{\mathsf{A}[\alpha,\beta]}{\sqrt{|\mathcal{O}(\alpha)|\,|\mathcal{O}(\beta)|}}\langle u|_{\mathcal{O}(\alpha)},1\rangle\langle v|_{\mathcal{O}(\beta)},1\rangle\\
&=a^T\widehat{\mathsf{A}}_Q\,b\qquad\text{where } a_\alpha:=\frac{\langle u|_{\mathcal{O}(\alpha)},1\rangle}{\sqrt{|\mathcal{O}(\alpha)|}},\ b_\alpha:=\frac{\langle v|_{\mathcal{O}(\alpha)},1\rangle}{\sqrt{|\mathcal{O}(\alpha)|}}\\
&\le\left\|\widehat{\mathsf{A}}_Q\right\|_2\|a\|\cdot\|b\|\\
&=\left\|\widehat{\mathsf{A}}_Q\right\|_2\sqrt{\sum_{\alpha\in\mathbb{N}^n_{q/2}}\frac{\langle u|_{\mathcal{O}(\alpha)},1\rangle^2}{|\mathcal{O}(\alpha)|}}\sqrt{\sum_{\alpha\in\mathbb{N}^n_{q/2}}\frac{\langle v|_{\mathcal{O}(\alpha)},1\rangle^2}{|\mathcal{O}(\alpha)|}}\\
&\le\left\|\widehat{\mathsf{A}}_Q\right\|_2\sqrt{\sum_{\alpha\in\mathbb{N}^n_{q/2}}\|u|_{\mathcal{O}(\alpha)}\|^2}\sqrt{\sum_{\alpha\in\mathbb{N}^n_{q/2}}\|u|_{\mathcal{O}(\alpha)}\|^2}\qquad\text{(by Cauchy-Schwarz)}\\
&\le\left\|\widehat{\mathsf{A}}_Q\right\|_2\|u\|\cdot\|v\|=\left\|\widehat{\mathsf{A}}_Q\right\|_2.
\end{aligned}$$

◀

The above two lemmas imply that $\|\mathsf{A}\|_2\le\|\widehat{\mathsf{A}}_Q\|_2\le\ 2^{O(q)}\cdot\frac{n^{q/4}}{q^{q/4}}$. Our moment matrix M is defined by

$$\mathsf{M}:=\frac{1}{c_1}\left(\frac{1}{c_2}\cdot\frac{q^{3q/4}}{n^{3q/4}}\mathsf{A}+\frac{\mathsf{W}}{n^{q/2}}\right),$$

where W is the direct extension of $\widehat{\mathsf{W}}$ constructed in Theorem 15 – $\mathsf{W}[I,J]:=\widehat{\mathsf{W}}[\alpha(I),\alpha(J)]$ for all $I,J\in[n]^{q/2}$, and $c_1,c_2=2^{\Theta(q)}$ that will be determined later.

We first consider the trace of M. The trace of A is 0 by design, and the trace of W is $n^{q/2}\cdot 2^{O(q)}$. Therefore, the trace of M can be made 1 by setting c_1 appropriately. Since both A and W are SoS-symmetric, so is M. Since $\mathbf{E}[\mathsf{W},A]=0$ and for each $I,J\in[n]^{q/2}$ with $i_1,\ldots,i_{q/2},j_1,\ldots,j_{q/2}$ all distinct we have $\mathbf{E}[\mathsf{A}[I,J]A[I,J]]=\frac{1}{q!}$, with high probability

$$\langle A,\mathsf{M}\rangle=\frac{1}{c_1}\cdot\langle A,\left(\frac{1}{c_2}\cdot\frac{q^{3q/4}}{n^{3q/4}}\mathsf{A}+\frac{\mathsf{W}}{n^{q/2}}\right)\rangle\ge 2^{O(-q)}\cdot\frac{q^{3q/4}}{n^{3q/4}}\cdot\frac{n^q}{q^q}=2^{O(-q)}\cdot\frac{n^{q/4}}{q^{q/4}}.$$

It finally remains to show that M is positive semidefinite. Take an arbitrary vector $v \in \mathbb{R}^{[n]^{q/2}}$, and let

$$p = \sum_{\alpha \in \mathbb{N}^n_{q/2}} x^\alpha p_\alpha = \sum_{\alpha \in \mathbb{N}^n_{q/2}} x^\alpha \cdot \left(\sum_{I \in [n]^{q/2} : \alpha(I) = \alpha} v_I \right)$$

be the associated polynomial. If $p = 0$, SoS-symmetry of M ensures $v\mathsf{M}v^T = 0$. Normalize v so that $\|p\|_{\ell_2} = 1$. First, consider another vector $v_m \in [n]^{q/2}$ such that

$$(v_m)_I = \begin{cases} \frac{p^{\alpha(I)}}{(q/2)!}, & \text{if } i_1, \dots, i_{q/2} \text{ are all distinct.} \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$\|v_m\|_2^2 \le \sum_{\alpha \in \mathbb{N}^n_{q/2}} p_\alpha^2/(q/2)! = \frac{1}{(q/2)!},$$

so $\|v_m\|_2 \le \frac{2^{O(q)}}{q^{q/4}}$. Since A is SoS-symmetric, has the minimum eigenvalue at least $-2^{O(q)} \cdot \frac{n^{q/4}}{q^{q/4}}$, and has nonzero entries only on the rows and columns $(i_1, \dots, i_{q/2})$ with all different entries,

$$v^T \mathsf{A} v = (v_m)^T \mathsf{A} (v_m) \ge 2^{-O(q)} \cdot \frac{n^{q/4}}{q^{3q/4}}.$$

We finally compute $v^T \mathsf{W} v$. Let $v_w \in [n]^{q/2}$ be the vector where for each $\alpha \in \mathbb{N}^n_{q/2}$, we choose one $I \in [n]^{q/2}$ arbitrarily and set $(v_w)_I = p_\alpha$ (all other $(v_w)_I$'s are 0). By SoS-symmetry of W,

$$v^T \mathsf{W} v = (v_w)^T \mathsf{W} (v_w) = p^T \widehat{\mathsf{W}} p \ge \frac{1}{2},$$

by Theorem 15. Therefore,

$$v^T \cdot \mathsf{M} \cdot v \;=\; \frac{1}{c_1} \cdot v^T \cdot \left(\frac{1}{c_2} \cdot \frac{q^{3q/4}}{n^{3q/4}} \mathsf{A} + \frac{\mathsf{W}}{n^{q/2}} \right) \cdot v \;\ge\; \frac{1}{c_1} \cdot \left(\frac{1}{c_2} \cdot 2^{-O(q)} \cdot \frac{n^{q/4}}{q^{3q/4}} \cdot \frac{q^{3q/4}}{n^{3q/4}} + \frac{1}{2} \cdot \frac{1}{n^{q/2}} \right) \;\ge\; 0,$$

by taking $c_2 = 2^{\Theta(q)}$. So M is positive semidefinite, and this finishes the proof of the lower bound in Theorem 1.

Thus we obtain,

▶ **Theorem 20** (Lower bound in Theorem 1). *For even $q \le n$, let $\mathcal{A} \in \mathbb{R}^{[n]^q}$ be a q-tensor with i.i.d. ±1 entries. Then with probability $1 - n^{\Omega(1)}$,*

$$\frac{\mathit{SoS}_q(\mathcal{A}(x))}{\mathcal{A}_{\max}} \;\ge\; \left(\frac{\Omega(n)}{q} \right)^{q/4 - 1/2}.$$

As a side note, observe that by applying Lemma 19 and the proof of Lemma 18 to the SoS-symmetric matrix representation of $f(x) = \mathcal{A}(x)$ (instead of A), we obtain a stronger SoS upper bound (by polylog factors) for the special case of $d = q$:

▶ **Theorem 21** (Upper bound in Theorem 1). *For even $q \le n$, let $\mathcal{A} \in \mathbb{R}^{[n]^q}$ be a q-tensor with i.i.d. ±1 entries. Then with probability $1 - n^{\Omega(1)}$,*

$$\frac{\mathit{SoS}_q(\mathcal{A}(x))}{\mathcal{A}_{\max}} \;\le\; \left(\frac{O(n)}{q} \right)^{q/4 - 1/2}.$$

Continuous Monitoring of ℓ_p Norms in Data Streams*

Jarosław Błasiok[1], Jian Ding[2], and Jelani Nelson[3]

1 **Harvard University, Cambridge, MA, USA**
`jblasiok@g.harvard.edu`
2 **University of Chicago, Chicago, MA, USA**
`jianding@galton.uchicago.edu`
3 **Harvard University, Cambridge, MA, USA**
`minilek@seas.harvard.edu`

Abstract

In insertion-only streaming, one sees a sequence of indices $a_1, a_2, \ldots, a_m \in [n]$. The stream defines a sequence of m frequency vectors $x^{(1)}, \ldots, x^{(m)} \in \mathbb{R}^n$ with $(x^{(t)})_i \stackrel{\text{def}}{=} |\{j : j \in [t], a_j = i\}|$. That is, $x^{(t)}$ is the frequency vector after seeing the first t items in the stream. Much work in the streaming literature focuses on estimating some function $f(x^{(m)})$. Many applications though require obtaining estimates at time t of $f(x^{(t)})$, for every $t \in [m]$. Naively this guarantee is obtained by devising an algorithm with failure probability $\ll 1/m$, then performing a union bound over all stream updates to guarantee that all m estimates are simultaneously accurate with good probability. When $f(x)$ is some ℓ_p norm of x, recent works have shown that this union bound is wasteful and better space complexity is possible for the continuous monitoring problem, with the strongest known results being for $p = 2$ [29, 10, 9]. In this work, we improve the state of the art for all $0 < p < 2$, which we obtain via a novel analysis of Indyk's p-stable sketch [30].

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases data streams, continuous monitoring, moment estimation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.32

1 Introduction

Estimating statistics of frequency vectors implicitly defined by insertion-only update streams, as defined in the abstract, was first studied by Flajolet and Martin in [24]. They studied the so-called *distinct elements problem*, in which $f(x)$ is the support size of x. In the insertion-only model, the support size of x is equivalent to the number of distinct a_i appearing in the stream. One goal in such streaming algorithms, both for this particular distinct elements problem as well as for many others function estimation problems studied in subsequent works, is to minimize the space consumption of the stream-processing algorithm, ideally using $o(n)$ words of memory (note there is always a trivial n space algorithm by storing x explicitly in memory).

For over two decades, work on estimating statistics of frequency vectors of streams remained dormant, until the work of [1] on estimating the p-norm $\|x\|_p = (\sum_{i=1}^n x_i^p)^{1/p}$ in

* J.B. supported by ONR grant N00014-15-1-2388. J.D. partially supported by NSF grant DMS-1455049 and an Alfred P. Sloan Research Fellowship. J.N. supported by NSF grant IIS-1447471 and CAREER award CCF-1350670, ONR Young Investigator award N00014-15-1-2388, and a Google Faculty Research Award.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 32; pp. 32:1–32:13
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

streams for integer $p \geq 1$. Since then several works have studied these and several other problems, from the perspective of both upper and lower bounds, including estimating $\|x\|_p$ for all $0 < p \leq 2$ (not necessarily integral) [1, 30, 32, 49, 41, 42, 39, 44, 38, 37], $\|x\|_p$ for $p > 2$ [1, 4, 21, 33, 7, 26, 34, 2, 16, 13, 25], empirical entropy [19, 20, 6, 28] and other information-theoretic quantities [31, 27, 14], cascaded norms [22, 36, 35], and several others. There have also been general theorems classifying which statistics of frequency vectors admit space-efficient streaming estimation algorithms [15, 8, 17, 12, 11].

Taking a dynamic data structural viewpoint, "streaming algorithms" is simply a synonym for "dynamic data structures" but with an implied focus on minimizing memory consumption (typically striving for an algorithm using *sublinear* memory). Elements in the stream can be viewed as updates to the frequency vector x (seeing $a \in [n]$ in the stream can be seen as $\texttt{update}(a, 1)$, causing the change $x_a \to x_a + 1$), and the request for an estimate of some statistic of x is a query. In this data structural language, all the works cited in the previous paragraph provide Monte-Carlo guarantees of the following form for queries: starting from any fixed frequency vector and after executing any fixed sequence of updates, the probability that the output of a subsequent query then fails is at most δ. Here we say a query fails if, say, the output is not a good approximation to some particular $f(x)$ (this will be made more formal later). In many applications however, one does not simply want the answer to one query at the end of some large number of updates, but rather one wants to *continuously* monitor the data stream. That is, the sequence of data structural operations is an intermingling of updates and queries. For example, one may have a threshold T in mind, and if $f(x)$ ever increases beyond T some data analyst should be alerted. Such a goal could be achieved (approximately) by querying after every update to determine whether the updated frequency vector satisfies this property. Indeed, the importance of supporting continuous queries in append-only databases (analogous to the insertion-only model of streaming) was recognized 25 years ago in [47], with several later works focused on continuous stream monitoring with application areas in mind such as trend detection, anomaly detection, financial data analysis, and (bio)sensor data analysis [3, 18, 46].

If one assumes that a query is being issued after every update, then in a stream of m updates the failure probability should be set to $\delta \ll 1/m$ so that, by a union bound, all queries succeed. Most Monte-Carlo streaming algorithms achieve some space S to achieve failure probability 1/3, at which point one can achieve failure probability δ by running $\Theta(\lg(1/\delta))$ instantiations of the algorithm in parallel and returning the median estimate (see for example [1]). This method increases the space from S to $\Theta(S \lg(1/\delta))$, and for many problems (such as ℓ_p-norm estimation) it is known that at least in the so-called strict turnstile model (i.e. $\texttt{update}(a, \Delta)$ is allowed for both positive and negative Δ, but we are promised $x_i \geq 0$ for all i at all times) this form of space blow-up is necessary [37]. Nevertheless, although improved space lower bounds have been given when desiring that the answer to a *single* query fails with probability at most δ, no such blow-up has been shown necessary for the continuous monitoring problem in which one wants, with failure probability 1/3, to provide simultaneously correct answers for m queries intermingled with m updates. In fact to the contrary, in certain scenarios such as estimating distinct elements or the ℓ_2-norm in insertion-only streams, improved *upper bounds* have been given!

▶ **Definition 1.** We say a Monte-Carlo randomized streaming algorithm $\mathcal{A}$ provides ***strong tracking*** for f in a stream of length m with failure probability η if at each time $t \in [m]$, $\mathcal{A}$ outputs an estimate $\tilde{f}_t$ such that

$$\mathbb{P}(\exists t \in [m] : |\tilde{f}^t - f(x^{(t)})| > \varepsilon f(x^{(t)})) < \eta.$$

We say that $\mathcal{A}$ provides ***weak tracking*** for f if

$$\mathbb{P}(\exists t \in [m] : |\tilde{f}^t - f(x^{(t)})| > \varepsilon \sup_{t' \in [m]} f(x^{(t')})) < \eta.$$

Note if f is monotonically increasing, then for insertion-only streams $\sup_{t'\in[m]} f(x^{(t')})$ is simply $f(x^{(m)})$.

The first non-trivial tracking result we are aware of which outperformed the median trick for insertion-only streaming was the RoughEstimator algorithm given in [40] for estimating the number of distinct elements in a stream. RoughEstimator provided a strong tracking guarantee for $f(x) = |\,support(x)|$ (the distinct elements problem) for constant ε, η, using the same space as what is what is required to answer only a single query. This strong tracking algorithm was used as a subroutine in the main *non-tracking* algorithm of that work for approximating the number of distinct elements in a data stream up to $1 + \varepsilon$.

For ℓ_p-estimation for $p \in (0, 2]$, without tracking, it is known that $\mathcal{O}(\varepsilon^{-2} \lg(1/\delta))$ words of memory is achievable to return a $(1 + \varepsilon)$-approximate value of $f(x) = \|x\|_p$ with failure probability δ [1, 30, 39][1]. This upper bound thus implies a strong tracking algorithm with space complexity $\mathcal{O}(\varepsilon^{-2} \lg m)$ for tracking failure probability $\eta = 1/3$, by setting $\delta < 1/(3m)$ and performing a union bound. The work [29] considered the strong tracking variant of ℓ_p-estimation in insertion-only streams for for any p in the more restricted interval $(1, 2]$. They showed that the same algorithms of [1, 30], unchanged, provide strong tracking with $\eta = 1/3$ with space $\mathcal{O}(\varepsilon^{-2}(\lg n + \lg\lg m + \lg(1/\varepsilon)))$ words[2]. This is an improvement over the standard median trick and union bound when the stream length is very long $(m > n^{\omega(1)})$ and ε is not too small $(\varepsilon > 1/m^{o(1)})$. They also showed that in an update model which allows deletions of items ("turnstile streaming"), any algorithm which only maintains a linear sketch Πx of x must use $\Omega(\lg m)$ words of memory for constant ε, showing that the median trick is optimal for this restricted class of algorithms.

A different algorithm was given in [10] for strong tracking for ℓ_2 using space $\mathcal{O}(\varepsilon^{-2}(\lg(1/\varepsilon)+ \lg\lg m))$. It was then most recently shown in [9] that the AMS sketch itself of [1] (though with 8-wise independent hash functions instead of the original 4-wise independence proposed in [1]) provides strong tracking in space $\mathcal{O}(\varepsilon^{-2} \lg\lg m)$, and weak tracking in space $\mathcal{O}(1/\varepsilon^2)$. That is, the AMS sketch provides weak tracking without any asymptotic increase in space complexity over the requirement to correctly answer only a single query.

Despite the progress in upper bounds for tracking ℓ_2, the only non-trivial improvement for tracking ℓ_p is the $\mathcal{O}(\varepsilon^{-2}(\lg n + \lg\lg m + \lg(1/\varepsilon)))$ upper bound of [29]. Although this bound provides an improvement for very long streams (m super-polynomial in n), it does not provide any improvement over the standard median trick for the case most commonly studied case in the literature of m, n being polynomially related.

Our contribution

We show that Indyk's p-stable sketch [30] for $0 < p \leq 2$, derandomized using bounded independence as in [39], provides weak tracking while using $\mathcal{O}(\lg(1/\varepsilon)/\varepsilon^2)$ words of space. It also provides strong tracking using $\mathcal{O}(\varepsilon^{-2}(\lg\lg m + \lg(1/\varepsilon))$ words of space. Our bounds thus both improve the space complexity achieved in [29] for ℓ_p-tracking, and well as the

[1] For constant δ and $p = 2$, [1] shows that space $\mathcal{O}(\varepsilon^{-2}(\lg n + \lg\lg m))$ *bits* is achievable in insertion-only streams.

[2] For $p = 2$ their space is as written including the space required to store all hash functions, but for $1 < p < 2$ this space bound assumes that the storage of hash functions is for free.

range of p supported from $p \in (1,2]$ to all $p \in (0,2]$ (note for $p > 2$, it is known that any algorithm requires polynomial space even to obtain a 2-approximation for a single query, i.e. the non-tracking variant of the problem [4]).

2 Notation

We use $[n]$ for integer n to denote $\{1,\dots,n\}$. We measure space in words unless stated otherwise, where a single word is at least $\lg(nm)$ bits. For $p \in (0,2]$, we let $\mathcal{D}_p$ denote the symmetric p-stable distribution, scaled so that for $Z \sim \mathcal{D}_p$, $\mathbb{P}(|Z| > 1) = \frac{1}{2}$. The distribution $\mathcal{D}_p$ has the property that it is supported on the reals, and for any fixed vector $v \in \mathbb{R}^n$ and $Z_1,\dots,Z_n,Z$ i.i.d. from $\mathcal{D}_p$, $\sum_{i=1}^n Z_i x_i$ is equal in distribution to $\|x\|_p \cdot Z$. See [45] for further reading on these distributions.

For two vectors $u, v \in \mathbb{R}^n$ we write $u \preceq v$ to denote coordinatewise comparison, i.e. $u \preceq v$ iff $\forall_i u_i \leq v_i$. For a finite set S, we write $\#S$ to denote cardinality of this set.

3 Preliminaries

The following lemma is standard. A proof with explicit constants can be found in [43, Theorem 42].

▶ **Lemma 2.** *If $Z \sim \mathcal{D}_p$, then $\mathbb{P}(Z > \lambda) \leq \frac{C_p}{\lambda^p}$ for some explicit constant C_p depending only on p.*

We also state some other results we will need.

▶ **Lemma 3** (Paley-Zygmund). *If $Z \geq 0$ is a random variable with finite variance, then*

$$\mathbb{P}(Z > \theta\,\mathbb{E}\,Z) \geq (1-\theta)^2 \frac{(\mathbb{E}\,Z)^2}{\mathbb{E}(Z^2)}.$$

▶ **Corollary 4.** *For fixed vector $v \in \mathbb{R}^n$, if $\sigma \in \{\pm 1\}^n$ is a vector of 4-wise independent random signs, then*

$$\mathbb{P}(\langle \sigma, v\rangle^2 \geq \frac{2}{3}\|v\|_2^2) \geq \frac{1}{27}\,.$$

Proof. This follows from $\mathbb{E}\langle \sigma, v\rangle^4 < 3(\mathbb{E}\langle \sigma, v\rangle^2)^2$ and the Paley-Zygmund inequality. ◀

▶ **Theorem 5** ([10, 9, Theorem 15]). *Let $v^{(1)}, v^{(2)}, \dots v^{(m)} \in \mathbb{R}^n$, be a sequence of vectors such that $0 \preceq v^{(1)} \preceq v^{(2)} \preceq \dots \preceq v^{(m)}$. Let $\sigma \in \{\pm 1\}^n$ be a vector of 4-wise independent random signs. Then*

$$\mathbb{P}\left(\sup_{i \leq m} |\langle \sigma, v^{(i)}\rangle| > \lambda \|v^{(n)}\|_2\right) < \frac{C}{\lambda^2}$$

for some universal constant C.

▶ **Theorem 6.** *[39, 23] If $Z_i \sim \mathcal{D}_p$ for $i \in [n]$ are k-wise independent random variables, then for every vector $x \in \mathbb{R}^n$ and every pair $a, b \in \mathbb{R} \cup \{\pm\infty\}$ we have*

$$\mathbb{P}(\langle Z, x\rangle \in (a,b)) = \mathbb{P}(\|x\|_p Z_1 \in (a,b)) \pm \mathcal{O}(k^{-1/p})\,.$$

▶ **Theorem 7.** *[5, Lemma 2.3] Let $X_1, \dots X_n \in \{0,1\}$ be a sequence of k-wise independent random variables, and let $\mu = \sum_{i=1}^n \mathbb{E}\,X_i$. Then*

$$\forall \lambda > 0,\ \mathbb{P}(\sum_{i=1}^n X_i \geq (1+\lambda)\mu) \leq \exp(-\Omega(\min\{\lambda, \lambda^2\}\mu)) + \exp(-\Omega(k))\,.$$

4 Overview of approach

Indyk's p-stable sketch picks a random matrix $\Pi \in \mathbb{R}^{d\times n}$ such that each entry is drawn according to the distribution $\mathcal{D}_p$. It then maintains the sketch $\Pi x^{(t)}$ of the current frequency vector. This sketch can be easily updated as the frequency vector changes, i.e. after observing an index $a_j \in [n]$ we update the sketch by $\Pi x^{(t+1)} := \Pi x^{(t)} + \Pi e_{a_j}$. An $\|x\|_p$-estimate query is answered by returning the median of $|\Pi x^{(i)}|_j$ over $j \in [d]$. Since storing Π in memory explicitly is prohibitively expensive, we generate it so that the entries in each row are k-wise independent for $k = \mathcal{O}(1/\varepsilon^p)$ (as done in [39]), and the d seeds used to generate the rows of Π are $\mathcal{O}(\lg(1/(\varepsilon\delta)))$-wise independent. We also work with discretized p-stable random variables to take bounded memory. All together, the bounded independence and discretization, also performed in [39], allow us to store Π using low memory.

We then show that instantiating Indyk's algorithm with $d = \mathcal{O}(\varepsilon^{-2}\lg(1/(\varepsilon\delta)))$ provides the weak tracking guarantee with failure probability δ. The analysis of the correctness of this algorithm is as follows. Let π_i denote the ith row of Π. We first show a result resembling the Doob's martingale inequality – namely, in Section 5 we show that for a fixed i, if we look at the evolution of $\langle \pi_i, x^{(t)}\rangle$ as t increases, the largest attained value ($\sup_{t\le m}\langle \pi_i, x^{(t)}\rangle$) is with good probability not much larger than the median of the distribution $|\langle \pi_i, x^{(m)}\rangle|$, which is the typical magnitude of the counter at the end of the stream. This fact resembles similar facts shown in [10, 9] for when the π_i have independent Rademachers as entries, though our situation is complicated by the fact that p-stable random variables have much heavier tails.

We then, discussed in Section 5.1, show how the previous paragraph implies a weak tracking algorithm with $d = \mathcal{O}(\varepsilon^{-2}\lg(1/(\varepsilon\delta)))$: we split the sequence of updates into $poly(1/\varepsilon)$ intervals such that the ℓ_p-norm of the frequency vector of updates in each of those intervals, i.e. $\|x^{(t+1)} - x^{(t)}\|_p$, is of the order $\varepsilon^{\Theta(1)}\|x^{(m)}\|_p$. We then union bound over the $poly(1/\varepsilon)$ intervals to argue that the algorithm's estimate is good at each of the interval endpoints. This is the source of the extra factor of $\lg(1/\varepsilon)$ in our space bound: to obtain $\varepsilon^{-\Omega(1)}$ failure probability to union bound over these intervals. On the other hand, within each of the intervals most of the counters do not change too rapidly by the argument developed in Section 5.

Finally, in Section 5.2 we show how given an algorithm satisfying a weak tracking guarantee, one can use it to get a strong-tracking algorithm with slightly larger space complexity. This argument was already present in [9]. One first identifies q points in the input stream at which the ℓ_p norm roughly doubles when compared to the previously marked point. There are only $\mathcal{O}(\lg m)$ such intervals. It is then enough to ensure that our algorithm satisfies weak tracking for all those $\mathcal{O}(\lg m)$ prefixes simultaneously, in order to deduce that the algorithm in fact satisfies strong tracking. This is done by union bound over $\mathcal{O}(\lg m)$ bad events (as opposed to standard union bound over $\mathcal{O}(m)$ bad events), which introduces an extra $\lg\lg m$ factor in the space complexity as when compared to weak tracking.

5 Analysis

We first show two lemmas that play a crucial role in our weak tracking analysis.

▶ **Lemma 8.** *Let $x \in \mathbb{R}^n$ be a fixed vector, and $Z \in \mathbb{R}^n$ be a random vector with k-wise independent entries drawn according to $\mathcal{D}_p$. Then*

$$\mathbb{P}(\sum_{i=1}^{n} x_i^2 Z_i^2 \ge \lambda^2\|x\|_p^2) \le \frac{C}{\lambda^p} + \mathcal{O}(k^{-1/p})$$

for some universal constant C.

Proof. Let E_0 be the event $\sum_{i=1}^n x_i^2 Z_i^2 \geq \lambda^2 \|x\|_p^2$. Note that E_0 depends only on $|Z_i|$, and does not depend on the signs of the Z_i. We write $Z_i = |Z_i|\sigma_i$, where σ_i are k-wise independent random signs. Conditioning on $|Z_i|$,

$$\mathop{\mathbb{E}}_{\sigma}\left(\left(\sum_{i=1}^n x_i |Z_i| \sigma_i\right)^2 \middle| |Z_1|, \ldots |Z_n|\right) = \sum_{i=1}^n x_i^2 Z_i^2$$

and therefore for any $|Z_1|, \ldots, |Z_m|$ for which E_0 holds, by Corollary 4

$$\begin{aligned}
&\mathop{\mathbb{P}}_{\sigma}\left(\left(\sum_{i=1}^n x_i |Z_i| \sigma_i\right)^2 \geq \frac{2}{3}\lambda^2 \|x\|_p^2 \middle| |Z_1|, \ldots, |Z_m|\right)\\
&\geq \mathop{\mathbb{P}}_{\sigma}\left(\left(\sum_{i=1}^n x_i |Z_i| \sigma_i\right)^2 \geq \frac{2}{3}\sum_{i=1}^n x_i^2 Z_i^2 \middle| |Z_1|, \ldots, |Z_m|\right)\\
&\geq \frac{1}{27}
\end{aligned}$$

and thus

$$\mathop{\mathbb{P}}_{\sigma}\left(\left(\sum_{i=1}^n x_i |Z_i| \sigma_i\right)^2 \geq \frac{2}{3}\lambda^2 \|x\|_p^2 \middle| |Z_1|, \ldots |Z_n|\right) \geq \frac{\mathbf{1}_{E_0}}{27},$$

where $\mathbf{1}_{E_0}$ is an indicator random variable for event E_0. Integrating over $|Z_i|$,

$$\mathop{\mathbb{P}}_{\sigma, Z}\left(\left(\sum_{i=1}^n x_i |Z_i| \sigma_i\right)^2 \geq \frac{2}{3}\lambda^2 \|x\|_p^2\right) \geq \frac{1}{27}\mathop{\mathbb{P}}_{Z}(E_0). \tag{1}$$

On the other hand $|Z_i|\sigma_i$ has the same distribution as Z_i, and moreover

$$\begin{aligned}
\mathop{\mathbb{P}}_{Z}\left(\left(\sum_{i=1}^n x_i Z_i\right)^2 \geq \frac{2}{3}\lambda^2 \|x\|_p^2\right) &= \mathop{\mathbb{P}}_{Z}\left(|\langle x, Z\rangle| \geq \sqrt{\frac{2}{3}}\lambda \|v\|_p\right)\\
&\leq \mathop{\mathbb{P}}_{Z}\left(\|x\|_p \tilde{Z} \geq \sqrt{\frac{2}{3}}\lambda \|x\|_p\right) + \mathcal{O}(k^{-1/p})\\
&\leq \frac{C}{\lambda^p} + \mathcal{O}(k^{-1/p})
\end{aligned} \tag{2}$$

where $\tilde{Z} \sim \mathcal{D}_p$. The inequalities are obtained via Theorem 6 and Lemma 2. Combining (1), (2) yields

$$\mathop{\mathbb{P}}_{Z}(E_0) \leq \frac{27C}{\lambda^p} + \mathcal{O}(k^{-1/p}). \qquad ◀$$

▶ **Lemma 9.** *Let $x^{(1)}, x^{(2)}, \ldots x^{(m)} \in \mathbb{R}^n$ satisfy $0 \preceq x^{(1)} \preceq x^{(2)} \preceq \ldots \preceq x^{(m)}$. Let $Z \in \mathbb{R}^n$ have k-wise independent entries marginally distributed according to $\mathcal{D}_p$. Then for some C_p depending only on p,*

$$\mathbb{P}\left(\sup_{k \leq m} |\langle Z, x^{(k)}\rangle| \geq \lambda \|x^{(m)}\|_p\right) \leq C_p\left(\frac{1}{\lambda^{2p/(2+p)}} + k^{-1/p}\right).$$

Proof. Observe that for any β we have

$$\mathbb{P}\left(\sup_{k\leq m}|\langle Z, x^{(k)}\rangle| \geq \lambda\|x^{(m)}\|_p\right) \leq \mathbb{P}\left(\sum_{i=1}^{n} Z_i^2 (x^{(m)})_i^2 \geq \beta^2\|x^{(m)}\|_p^2\right) \\ + \mathbb{P}\left(\sup_{k\leq m}|\langle Z, x^{(k)}\rangle| \geq \lambda\|x^{(m)}\|_p \middle| \sum_{i=1}^{n} Z_i^2 (x^{(m)})_i^2 < \beta^2\|x^{(m)}\|_p^2\right).$$

Lemma 8 directly implies that

$$\mathbb{P}\left(\sum_{i=1}^{n} Z_i^2 (x^{(m)})_i^2 \geq \beta^2\|x^{(m)}\|_p^2\right) \leq \frac{C}{\beta^p} + \frac{C}{k^{1/p}}. \tag{3}$$

On the other hand we can write $Z_i = |Z_i|\sigma_i$, where σ_i are k-wise independent Rademacher random variables, independent from $|Z_i|$. Let us define $w^{(k)} \in \mathbb{R}^n$ for $k \in [m]$ to be the vector with coordinates $(w^{(k)})_i := (x^{(k)})_i|Z_i|$, so that $\langle x^{(k)}, Z\rangle = \langle w^{(k)}, \sigma\rangle$, and in particular

$$\sup_{k\leq m}\left|\langle Z, x^{(i)}\rangle\right| = \sup_{k\leq m}\left|\langle \sigma, w^{(i)}\rangle\right|.$$

Now, if we condition on $|Z_1|, \ldots |Z_n|$, then the sequence $w^{(1)}, \ldots w^{(k)}$ of vectors satisfies the assumptions of Theorem 5, and we can conclude that

$$\mathbb{P}\left(\sup_{k\leq m}\left|\langle \sigma, w^{(k)}\rangle\right| > \frac{\lambda}{\beta}\|w^{(m)}\|_2\right) \leq \frac{C\beta^2}{\lambda^2}.$$

Moreover if $|Z_i|$ are such that $\sum_{i=1}^n Z_i^2 (x^{(m)})_i^2 \leq \beta^2\|x^{(m)}\|_p^2$, or equivalently $\|w^{(m)}\|_2^2 \leq \beta^2\|x^{(m)}\|_p^2$, we have

$$\mathbb{P}\left(\sup_{k\leq m}\left|\langle \sigma, w^{(k)}\rangle\right| > \lambda\|x^{(m)}\|_p\right) \leq \frac{C\beta^2}{\lambda^2},$$

which implies

$$\mathbb{P}\left(\sup_{k\leq m}|\langle Z, x^{(k)}\rangle| \geq \lambda\|x^{(m)}\|_p \middle| \sum_{i=1}^{n}(Z_i x_i^{(m)})^2 < \beta\|x^{(m)}\|_p^2\right) \leq \frac{C\beta^2}{\lambda^2}.$$

This together with Equation (3) yields

$$\mathbb{P}\left(\sup_{k\leq m}|\langle Z, x^{(k)}\rangle| \geq \lambda\|x^{(m)}\|_p\right) \leq \frac{1}{\beta^p} + \frac{C\beta^2}{\lambda^2} + \frac{C}{k^{1/p}}.$$

We can take $\beta := \Theta(\lambda^{\frac{2}{2+p}})$, to have $\frac{1}{\beta^p} + \frac{C\beta^2}{\lambda^2} = \mathcal{O}(\lambda^{-\frac{2p}{2+p}})$. ◀

5.1 Weak tracking of $\|x\|_p$

In this section we upper bound the number of rows needed in Indyk's p-stable sketch with boundedly independent entries to achieve weak tracking.

▶ **Lemma 10.** *Let $x^{(1)}, \ldots x^{(m)} \in \mathbb{R}^n$ be any sequence satisfying $0 \preceq x^{(1)} \preceq x^{(2)} \preceq \ldots \preceq x^{(m)}$. Take $\Pi \in \mathbb{R}^{d\times n}$ to be a random matrix with entries drawn according to $\mathcal{D}_p$, and such that the rows are r-wise independent, and all entries within a row are s-wise independent.*

For every $k \in [m]$, define s_k to be median $\left(|(\Pi x^{(k)})_1|, \ldots, |(\Pi x^{(k)})_d|\right)$. If $d = \Omega(\varepsilon^{-2}(\lg\frac{1}{\varepsilon} + \lg\frac{1}{\delta}))$, $r = \Omega(\lg\frac{1}{\varepsilon} + \lg\frac{1}{\delta})$ and $s = \Omega(\varepsilon^{-p})$, then with probability at least $1-\delta$ we have

$$\forall k \in [m],\ \|x^{(k)}\|_p - \varepsilon\|x^{(m)}\|_p \leq s_k \leq \|x^{(k)}\|_p + \varepsilon\|x^{(m)}\|_p.$$

Proof. Consider a sequence of indices $1 < t_1 < t_2 < \ldots < t_{q+1} = m$, constructed inductively in the following way. We take t_1 to be the smallest index with $\|x^{(t_1)}\|_p \geq \varepsilon^4 \|x^{(m)}\|_p$. Given t_k, we take t_{k+1} to be the smallest index such that $\|x^{(t_{k+1})} - x^{(t_k)}\|_p \geq \varepsilon^4 \|x^{(m)}\|_p$ if there exists one, and $t_{k+1} = m$ otherwise.

Observe $q \leq \varepsilon^{-8}$. Indeed, for $p \geq 1$ we have

$$\|x^{(m)}\|_p^p = \|x^{(t_1)} + \sum_{1 \leq i < q} (x^{(t_{i+1})} - x^{(t_i)})\|_p^p \geq \|x^{(t_1)}\|_p^p + \sum_{1 \leq i < q} \|x^{(t_{i+1})} - x^{(t_i)}\|_p^p \geq q\varepsilon^{4p} \|x^{(m)}\|_p^p$$

where the inequality $\|x^{(t_1)} + \sum_{i \geq 1}(x^{(t_{i+1})} - x^{(t_i)})\|_p^p \geq \|x^{(t_1)}\|_p^p + \sum_{1 \leq i < q} \|x^{(t_{i+1})} - x^{(t_i)}\|_p^p$ holds because all vectors $x^{(1)}$ and $x^{(t_{i+1})} - x^{(t_i)}$ for every i have non-negative entries – we can consider each coordinate separately, and use the fact that for $p \geq 1$ and nonnegative numbers a_i we have $(\sum a_i)^p \geq \sum a_i^p$ – or equivalently, $\|a\|_1^p \geq \|a\|_p^p$. After rearranging this yields $q \leq \varepsilon^{-4p}$.

Similarly, for $p \leq 1$, we have that for non-negative numbers a_i, $(\sum_{i \leq q} a_i)^p \geq q^{p-1} \sum i \leq q a_i^p$ (this is true because for fixed $\sum a_i$, the sum $\sum a_i^p$ is maximized when all a_i are equal), and therefore

$$\begin{aligned}\|x^{(m)}\|_p^p = \|x^{(t_1)} + \sum_{1 \leq i < q} (x^{(t_{i+1})} - x^{(t_i)})\|_p^p &\geq q^{p-1} \left(\|x^{(t_1)}\|_p^p + \sum_{1 \leq i < q} \|x^{(t_{i+1})} - x^{(t_i)}\|_p^p \right) \\ &\geq q^p \varepsilon^{4p} \|x^{(m)}\|_p^p\end{aligned}$$

which implies $q \leq \varepsilon^{-4}$.

For $j \in [m]$, let us define

$$\begin{aligned} l_j &:= \#\{i : |\langle \pi_i, x^{(j)} \rangle| < (1-\varepsilon)\|x^{(j)}\|_p\} \\ u_j &:= \#\{i : |\langle \pi_i, x^{(j)} \rangle| > (1+\varepsilon)\|x^{(j)}\|_p\}. \end{aligned}$$

Let $\tilde{\pi}_i$ be a vector of i.i.d. random variables drawn according to $\mathcal{D}_p$. We know that $\langle \tilde{\pi}_i, x^{(j)} \rangle \sim \|x^{(j)}\|_p \mathcal{D}_p$. Hence $\mathbb{P}(|\langle \tilde{\pi}_i, x^{(j)} \rangle| > \|x^{(j)}\|_p) = \frac{1}{2}$, and $\mathbb{P}(|\langle \tilde{\pi}_i, x^{(j)} \rangle| > (1+\varepsilon)\|x^{(j)}\|_p) \leq \frac{1}{2} - 2C\varepsilon$ for some universal constant C. Similarly $\mathbb{P}(|\langle \tilde{\pi}_i, x^{(j)} \rangle| < (1-\varepsilon)\|x^{(j)}\|_p) \leq \frac{1}{2} - 2C\varepsilon$.

Entries of π_i are s-wise independent, for $s \geq C_2 \varepsilon^{-p}$ with some large constant C_2 depending on C. Thus by Theorem 6, $\mathbb{P}(|\langle \pi_i, x^{(j)} \rangle| < (1-\varepsilon)\|x^{(j)}\|_p) \leq \mathbb{P}(|\langle \tilde{\pi}_i, x^{(j)} \rangle| < (1-\varepsilon)\|x^{(j)}\|_p) + C\varepsilon \leq \frac{1}{2} - C\varepsilon$, and analogously for $\mathbb{P}(|\langle \pi_i, x^{(j)} \rangle| > (1+\varepsilon)\|x^{(j)}\|_p) < \frac{1}{2} - C\varepsilon$.

Hence

$$\begin{aligned} \mathbb{E}\, l_j &\leq d\left(\frac{1}{2} - C\varepsilon\right) \\ \mathbb{E}\, u_j &\leq d\left(\frac{1}{2} - C\varepsilon\right). \end{aligned}$$

For $j \in [q]$, let S_j be the event

$$\left\{ l_{t_j} \leq \frac{d}{2} - \frac{Cd}{2}\varepsilon \right\} \wedge \left\{ u_{t_j} \leq \frac{d}{2} - \frac{Cd}{2}\varepsilon \right\}$$

Note that for fixed j and varying i, indicator random variables for the events "$|\langle \pi_i, x^{(j)} \rangle| < (1-\varepsilon)\|x^{(j)}\|_p$" are r-wise independent. Thus by Theorem 7, $\mathbb{P}(S_j) \geq 1 - C' \exp(-\Omega(d\varepsilon^2)) - \exp(-\Omega(r))$. Taking $d = \Omega(\varepsilon^{-2}(\lg \frac{1}{\varepsilon} + \lg \frac{1}{\delta}))$ and $r = \Omega(\lg \frac{1}{\varepsilon\delta})$ we obtain $\mathbb{P}(S_j) \geq 1 - \frac{\delta\varepsilon^8}{2}$, and hence by a union bound all S_j hold simultaneously except with probability at most $\frac{\delta}{2}$ since the number of events S_j is $q \leq \varepsilon^{-8}$.

For $i \in [d]$ and $j \in [q]$, let $E_{i,j}$ be the event

$$\exists s \in [t_j, t_{j+1} - 1],\ |\langle x^{(s)} - x^{(t_j)}, \pi_i \rangle| > \varepsilon \|x^{(m)}\|_p.$$

By construction of the sequence t_j, all $x^{(s)} - x^{(t_j)}$ above have ℓ_p norm at most $\varepsilon^4 \|x^{(m)}\|_p$, we can invoke Lemma 9 to deduce that $\mathbb{P}(E_{ij}) \leq C_3 \left(\frac{\varepsilon^4}{\varepsilon}\right)^{2/3} + C_3 s^{-1/p}$. Again if we pick $s \geq C_4 \varepsilon^{-p}$ for sufficiently large C_4 and small enough ε we have $\mathbb{P}(E_{ij}) \leq \frac{C}{4}\varepsilon$. Therefore for any fixed j, we have

$$\mathbb{E} \sum_{i=1}^{d} \mathbf{1}_{E_{ij}} \leq \frac{C}{4} d\varepsilon$$

And finally again by Theorem 7, for each j

$$\mathbb{P}(\sum_{i=1}^{d} \mathbf{1}_{E_{ij}} \geq \frac{C}{2} d\varepsilon) \lesssim \exp(-C' d\varepsilon) + \exp(-C' r)$$

We have $d \geq C_3 \varepsilon^{-2} \lg \frac{1}{\delta\varepsilon}$, and $q \leq \varepsilon^{-8}$, hence for sufficiently small ε, we have $\exp(-C' d\varepsilon) \leq \frac{\delta}{2q}$. On the other hand if $r = \Omega(\lg \frac{1}{\delta\varepsilon})$ is sufficiently large, we have $\exp(-C' r) \leq \frac{\delta}{2q}$. We invoke the union bound over all j to deduce that with probability at least $1 - \frac{\delta}{2}$ the following event V holds:

$$\forall j,\ \sum_{i=1}^{d} \mathbf{1}_{E_{ij}} \leq \frac{C}{2} d\varepsilon.$$

We know that with probability at least $1 - \delta$ simultaneously V and all the events S_j hold. We will show now that, when these events all hold, then $\forall k$ $\|x^{(k)}\|_p - K\varepsilon\|x^{(m)}\|_p \leq s_k \leq \|x^{(k)}\|_p + K\varepsilon\|x^{(m)}\|_p$ for some universal constant K. Indeed, consider some k, and let us assume that $t_j \leq k \leq t_{j+1}$. With event S_j satisfied, we know that $\#\{i : |\langle \pi_i, x^{(t_j)} \rangle| \leq \|x^{(t_j)}\|_p + \varepsilon\|x^{(m)}\|_p\} \geq d\left(\frac{1}{2} + \frac{C\varepsilon}{2}\right)$, and with event V satisfied, we know that for all but $\frac{C\varepsilon}{2}d$ of indices i we have $|\langle \pi_i, x^{(k)} - x^{(t_j)} \rangle| \leq \varepsilon\|x^{(m)}\|$.

By the triangle inequality $|\langle \pi_i, x^{(k)} \rangle| \leq |\langle \pi_i, x^{(t_j)} \rangle| + |\langle \pi_i, x^{(k)} - x^{(t_j)} \rangle|$, yielding

$$\#\{i : |\langle \pi_i, v_k \rangle| \leq \|v_{t_j}\|_p + 2\varepsilon\|v_m\|_p\} \geq \frac{d}{2}.$$

With similar reasoning we can deduce that

$$\#\{i : |\langle \pi_i, x^{(k)} \rangle| \geq \|x^{(t_j)}\|_p - 2\varepsilon\|x^{(m)}\|_p\} \geq \frac{d}{2},$$

which implies the median of $|\langle \pi_i, x^{(k)} \rangle|$ over $i \in [d]$ is in the range $\|x^{(t_j)}\|_p \pm 2\varepsilon\|x^{(m)}\|_p$. In other words

$$\|x^{(t_j)}\|_p - 2\varepsilon\|x^{(m)}\|_p \leq s_k \leq \|x^{(t_i)}\|_p + 2\varepsilon\|x^{(m)}\|_p.$$

Finally we also have $\big| \|x^{(k)}\|_p - \|x^{(t_j)}\|_p \big| \leq \varepsilon\|x^{(m)}\|_p$ by construction of the sequence $\{t_j\}_{j=1}^{q}$, so the claim follows up to rescaling ε by a constant factor. ◀

▶ **Lemma 11.** *The above algorithm can be implemented using $\mathcal{O}(\varepsilon^{-2} \lg(1/(\varepsilon\delta)) \lg m)$ bits of memory to store fixed precision approximations of all counters $(\Pi x^{(k)})_i$, and $\mathcal{O}(\varepsilon^{-p} \lg(1/(\varepsilon\delta)) \lg(nm))$ bits to store Π.*

Proof. Consider a sketch matrix Π as in Lemma 10 – i.e. $\Pi \in \mathbb{R}^{d\times n}$ with random $\mathcal{D}_p$ entries, such that all rows are r-wise independent and all entries within a row are s-wise independent. Moreover let us pick some $\gamma = \Theta(\varepsilon m^{-1})$ and consider discretization $\tilde{\Pi}$ of Π, namely each entry $\tilde{\Pi}_{ij}$ is equal to Π_{ij} rounded to the nearest integer multiple of γ. The analysis identical to the one in [39, A.6] shows that this discretization have no significant effect on the accuracy of the algorithm, and moreover that one can sample from a nearby distribution using only $\tau = \mathcal{O}(\lg m\varepsilon^{-1})$ uniformly random bits. Therefore we can store such a matrix succinctly using $\mathcal{O}\left(rs(\lg n + \tau) + r\lg d\right)$ bits of memory, by storing a seed for a random r-wise independent hash function $h : [d] \to \{0,1\}^{\mathcal{O}(s(\lg n + \tau))}$ and interpreting each $h(i)$ as a seed for an s-wise independent hash function describing the i-th row of $\tilde{\Pi}$ [48, Corollary 3.34]. Hence the total space complexity of storing the sketch matrix $\tilde{\Pi}$ in a succinct manner is $\mathcal{O}\left(\frac{\lg\delta^{-1}+\lg\varepsilon^{-1}}{\varepsilon^p}(\lg n + \lg m)\right)$ bits.

Additionally we have to store the sketch of the current frequency vector itself, i.e. for all $i \in [d]$ we need to store $\langle \tilde{\pi}_i, x^{(k)}\rangle$; for every such counter we need $\mathcal{O}(\lg m\varepsilon^{-1}) = \mathcal{O}(\lg m)$ bits, and there are $d = \mathcal{O}\left(\frac{\lg\varepsilon^{-1}+\lg\delta-1}{\varepsilon^{-2}}\right)$ counters. ◀

We thus have the following main theorem of this section.

▶ **Theorem 12.** *For any $p \in (0,2]$ there is an insertion-only streaming algorithm that provides the weak tracking guarantees for $f(x) = \|x\|_p$ with probability $1-\delta$ using at most $\mathcal{O}\left(\frac{\lg m+\lg n}{\varepsilon^2}(\lg\varepsilon^{-1} + \lg\delta^{-1})\right)$ bits of memory.*

5.2 Strong tracking of $\|x\|_p$

In this section we discuss achieving a strong tracking guarantee. The same argument for ℓ_2-tracking appeared in [9]. The reduction is in fact general, and shows that for any monotone function f the strong tracking problem for f reduces to the weak tracking version of the same problem with smaller failure probability.

▶ **Lemma 13.** *Let $f : \mathbb{R}^n \to \mathbb{R}_+$ be any monotone functon of $\mathbb{R}^n$ (i.e. $x \preceq y \implies f(x) \le f(y)$), such that $\min_i f(e_i) = 1$ (where e_i are standard basis vectors). Let $\mathcal{A}$ be an insertion-only streaming algorithm satisfying weak tracking for any sequence of updates with probability $1-\delta$ and accuracy ε. Then for a sequence of frequency vectors $0 \preceq x^{(1)} \preceq \ldots \preceq x^{(m)}$ algorithm $\mathcal{A}$ satisfies strong tracking with probability $1 - \delta\lg f(x^{(m)})$ and accuracy 2ε.*

Proof. Define $t_1 < t_2 < \cdots < t_q$ so that t_i is the smallest index in $[m]$ larger than t_{i-1} with $f(x^{(t_i)}) \ge 2^i$ (if no such index exists, define $q = i$ and $t_q = m$). Note that $q \le \lg f(x^{(m)})$.

The algorithm will fail with probability at most δ to satisfy the conclusion of Theorem 12 for a particular sequence of vectors $x^{(1)}, x^{(2)}, \ldots x^{(t_j)}$. That is, for every j, with probability $1-\delta$, we have that

$$\forall i \le t_j,\ f(x^{(i)}) - \varepsilon f(x^{(t_j)}) \le \tilde{f}^i \le f(x^{(i)}) + \varepsilon f(x^{(t_j)}),$$

where $\tilde{f}^t$ is the estimate output by the algorithm at time t.

We can union bound over all $j \in [q]$ to deduce that except with probability $q\delta \le \delta\lg f(x^{(m)})$,

$$\forall i \le t_j,\ f(x^{(i)}) - \varepsilon f(x^{(t_j)}) \le \tilde{f}^i \le f(x^{(i)}) + \varepsilon f(x^{(t_j)}).$$

By construction of the sequence of t_j, we know that for every i, if we take t_j to be smallest such that $i \le t_j$, then $f(x^{(t_j)}) \le 2f(x^{(i)})$, and the claim follows. ◀

▶ **Theorem 14.** *For any $p \in (0,2]$ there is an insertion-only streaming algorithm that provides strong tracking guarantees for estimating the ℓ_p-norm of the frequency vector with probability $1-\delta$ and multiplicative error $1+\varepsilon$, with space usage in bits bounded by $\mathcal{O}\left(\frac{\lg m + \lg n}{\varepsilon^2}(\lg \varepsilon^{-1} + \lg \delta^{-1} + \lg \lg m)\right)$.*

Proof. This follows from Lemma 11 and Lemma 13 by observing that after a sequence of m insertions, the ℓ_p norm of the frequency vector is bounded by m^2, i.e. $\lg(\|x^{(m)}\|_p) = \mathcal{O}(\lg m)$. ◀

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 *Proc. of the 52nd IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 363–372, 2011.

3 Shivnath Babu and Jennifer Widom. Continuous queries over data streams. *SIGMOD Record*, 30(3):109–120, 2001.

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 Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *Proc. of the 35th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 276–287, 1994.

6 Lakshminath Bhuvanagiri and Sumit Ganguly. Estimating entropy over data streams. In *Proc. of the 14th Annual European Symposium on Algorithms (ESA)*, pages 148–159, 2006.

7 Lakshminath Bhuvanagiri, Sumit Ganguly, Deepanjan Kesh, and Chandan Saha. Simpler algorithm for estimating frequency moments of data streams. In *Proc. of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 708–713, 2006.

8 Vladimir Braverman and Stephen R. Chestnut. Universal sketches for the frequency negative moments and other decreasing streaming sums. In *Proc. of the 18th International Workshop on Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques (APPROX)*, pages 591–605, 2015.

9 Vladimir Braverman, Stephen R. Chestnut, Nikita Ivkin, Jelani Nelson, Zhengyu Wang, and David P. Woodruff. BPTree: an ℓ_2 heavy hitters algorithm using constant memory. In *Proc. of the 36th SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, 2017.

10 Vladimir Braverman, Stephen R. Chestnut, Nikita Ivkin, and David P. Woodruff. Beating countsketch for heavy hitters in insertion streams. In *Proc. of the 48th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 740–753, 2016.

11 Vladimir Braverman, Stephen R. Chestnut, Robert Krauthgamer, and Lin F. Yang. Streaming symmetric norms via measure concentration. In *Proc. of the 49th Annual ACM Symposium on Theory of Computing (STOC), to appear*, 2017.

12 Vladimir Braverman, Stephen R. Chestnut, David P. Woodruff, and Lin F. Yang. Streaming space complexity of nearly all functions of one variable on frequency vectors. In *Proc. of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS)*, pages 261–276, 2016.

13 Vladimir Braverman, Jonathan Katzman, Charles Seidell, and Gregory Vorsanger. An optimal algorithm for large frequency moments using $o(n^{1-2/k})$ bits. In *Proc. of the 17th International Workshop on Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques (APPROX)*, pages 531–544, 2014.

14 Vladimir Braverman and Rafail Ostrovsky. Measuring independence of datasets. In *Proc. of the 42^{nd} ACM Symposium on Theory of Computing (STOC)*, pages 271–280, 2010.

15 Vladimir Braverman and Rafail Ostrovsky. Zero-one frequency laws. In *Proc. of the 42^{nd} ACM Symposium on Theory of Computing (STOC)*, pages 281–290, 2010.

16 Vladimir Braverman and Rafail Ostrovsky. Approximating large frequency moments with pick-and-drop sampling. In *Proc. of the 16^{th} International Workshop on Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques (APPROX)*, pages 42–57, 2013.

17 Vladimir Braverman, Rafail Ostrovsky, and Alan Roytman. Zero-one laws for sliding windows and universal sketches. In *Proc. of the 18^{th} International Workshop on Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques (APPROX)*, pages 573–590, 2015.

18 Donald Carney, Ugur Çetintemel, Mitch Cherniack, Christian Convey, Sangdon Lee, Greg Seidman, Michael Stonebraker, Nesime Tatbul, and Stanley B. Zdonik. Monitoring streams - A new class of data management applications. In *Proc. of the 28^{th} International Conference on Very Large Data Bases (VLDB)*, pages 215–226, 2002.

19 Amit Chakrabarti, Khanh Do Ba, and S. Muthukrishnan. Estimating entropy and entropy norm on data streams. *Internet Mathematics*, 3(1):63–78, 2006.

20 Amit Chakrabarti, Graham Cormode, and Andrew McGregor. A near-optimal algorithm for estimating the entropy of a stream. *ACM Trans. Algorithms*, 6(3):51:1–51:21, 2010.

21 Amit Chakrabarti, Subhash Khot, and Xiaodong Sun. Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In *Proc. of the 18^{th} Annual IEEE Conference on Computational Complexity (CCC)*, pages 107–117, 2003.

22 Graham Cormode and S. Muthukrishnan. Space efficient mining of multigraph streams. In *Proc. of the 24^{th} ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, pages 271–282, 2005.

23 Ilias Diakonikolas, Daniel M. Kane, and Jelani Nelson. Bounded independence fools degree-2 threshold functions. In *Proc. of the 51^{st} Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 11–20, 2010.

24 Philippe Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, 1985.

25 Sumit Ganguly. Taylor polynomial estimator for estimating frequency moments. In *Proc. of the 42^{nd} International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 542–553, 2015.

26 André Gronemeier. Asymptotically optimal lower bounds on the nih-multi-party information complexity of the and-function and disjointness. In *Proc. of the 26^{th} International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 505–516, 2009.

27 Sudipto Guha, Piotr Indyk, and Andrew McGregor. Sketching information divergences. *Machine Learning*, 72(1-2):5–19, 2008.

28 Nicholas J. A. Harvey, Jelani Nelson, and Krzysztof Onak. Sketching and streaming entropy via approximation theory. In *Proc. of the 49^{th} Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 489–498, 2008.

29 Zengfeng Huang, Wai Ming Tai, and Ke Yi. Tracking the frequency moments at all times. *CoRR*, abs/1412.1763, 2014.

30 Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, May 2006.

31 Piotr Indyk and Andrew McGregor. Declaring independence via the sketching of sketches. In *Proc. of the 19^{th} Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 737–745, 2008.

32 Piotr Indyk and David P. Woodruff. Tight lower bounds for the distinct elements problem. In *Proc. of the 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 283–, 2003.

33 Piotr Indyk and David P. Woodruff. Optimal approximations of the frequency moments of data streams. In *Proc. of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 202–208, 2005.

34 T. S. Jayram. Hellinger strikes back: A note on the multi-party information complexity of AND. In *Proc. of the 12th International Workshop on Randomization and Approximation Techniques (RANDOM)*, pages 562–573, 2009.

35 T. S. Jayram. On the information complexity of cascaded norms with small domains. In *IEEE Information Theory Workshop (ITW)*, pages 1–5, 2013.

36 T. S. Jayram and David P. Woodruff. The data stream space complexity of cascaded norms. In *Proc. of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 765–774, 2009.

37 T. S. Jayram and David P. Woodruff. Optimal bounds for johnson-lindenstrauss transforms and streaming problems with subconstant error. *ACM Trans. Algorithms*, 9(3):26:1–26:17, 2013.

38 Daniel M. Kane, Jelani Nelson, Ely Porat, and David P. Woodruff. Fast moment estimation in data streams in optimal space. In *Proc. of the 43rd ACM Symposium on Theory of Computing (STOC)*, pages 745–754, 2011.

39 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *Proc. of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1161–1178, 2010.

40 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. An optimal algorithm for the distinct elements problem. In *Proc. of the 29th SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 41–52, 2010.

41 Ping Li. Estimators and tail bounds for dimension reduction in ℓ_α ($0 < \alpha \leq 2$) using stable random projections. In *Proc. of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 10–19, 2008.

42 Ping Li. Compressed counting. In *Proc. of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 412–421, 2009.

43 Jelani Nelson. *Sketching and streaming high-dimensional vectors.* PhD thesis, Massachusetts Institute of Technology, 2011.

44 Jelani Nelson and David P. Woodruff. Fast manhattan sketches in data streams. In *Proc. of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 99–110, 2010.

45 J. P. Nolan. *Stable Distributions – Models for Heavy Tailed Data.* Birkhauser, Boston, 2017. In progress, Chapter 1 online at `http://fs2.american.edu/jpnolan/www/stable/stable.html`.

46 Chris Olston, Jing Jiang, and Jennifer Widom. Adaptive filters for continuous queries over distributed data streams. In *Proc. of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 563–574, 2003.

47 Douglas B. Terry, David Goldberg, David A. Nichols, and Brian M. Oki. Continuous queries over append-only databases. In *Proc. of the ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 321–330, 1992.

48 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012. `doi:10.1561/0400000010`.

49 David P. Woodruff. Optimal space lower bounds for all frequency moments. In *Proc. of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 167–175, 2004.

Vertex Isoperimetry and Independent Set Stability for Tensor Powers of Cliques[*][†]

Joshua Brakensiek

Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA, USA
jbrakens@andrew.cmu.edu

Abstract

The tensor power of the clique on t vertices (denoted by K_t^n) is the graph on vertex set $\{1,\ldots,t\}^n$ such that two vertices $x, y \in \{1,\ldots,t\}^n$ are connected if and only if $x_i \neq y_i$ for all $i \in \{1,\ldots,n\}$. Let the density of a subset S of K_t^n to be $\mu(S) := \frac{|S|}{t^n}$. Also let the vertex boundary of a set S to be the vertices of the graph, including those of S, which are incident to some vertex of S. We investigate two similar problems on such graphs.

First, we study the vertex isoperimetry problem. Given a density $\nu \in [0,1]$ what is the smallest possible density of the vertex boundary of a subset of K_t^n of density ν? Let $\Phi_t(\nu)$ be the infimum of these minimum densities as $n \to \infty$. We find a recursive relation allows one to compute $\Phi_t(\nu)$ in time polynomial to the number of desired bits of precision.

Second, we study given an independent set $I \subseteq K_t^n$ of density $\mu(I) = \frac{1}{t}(1-\epsilon)$, how close it is to a maximum-sized independent set J of density $\frac{1}{t}$. We show that this deviation (measured by $\mu(I \setminus J)$) is at most $4\epsilon^{\frac{\log t}{\log t - \log(t-1)}}$ as long as $\epsilon < 1 - \frac{3}{t} + \frac{2}{t^2}$. This substantially improves on results of Alon, Dinur, Friedgut, and Sudakov (2004) and Ghandehari and Hatami (2008) which had an $O(\epsilon)$ upper bound. We also show the exponent $\frac{\log t}{\log t - \log(t-1)}$ is optimal assuming n tending to infinity and ϵ tending to 0. The methods have similarity to recent work by Ellis, Keller, and Lifshitz (2016) in the context of Kneser graphs and other settings.

The author hopes that these results have potential applications in hardness of approximation, particularly in approximate graph coloring and independent set problems.

1998 ACM Subject Classification G.2.2 Graph Theory

Keywords and phrases extremal combinatorics, independent sets, isoperimetry, stability

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.33

1 Introduction

A growing subfield in extremal combinatorics is understanding the structure of combinatorial objects which are close in size to the maximal such objects. In this work, we study such questions in the context of independent sets of tensor power of cliques. We establish this by first understanding the isoperimetric properties of such graphs.

1.1 Vertex isoperimetry

For any undirected graph $G = (V_G, E_G)$ and $S \subseteq V_G$, we define the *vertex boundary* of S to be

$$\partial S := \{x \in V_G \,:\, \text{exists } y \in S \text{ such that } \{x,y\} \in E_G\}.$$

[*] Full version available at [6], http://arxiv.org/abs/1702.04432.

[†] This work was partially supported by REU supplements to NSF CCF-1526092 and CCF-1422045.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 33; pp. 33:1–33:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Furthermore, we define the *density* of S to be

$$\mu(S) := \frac{|S|}{|V_G|}.$$

The relationship between $\mu(S)$ and $\mu(\partial S)$ is typically captured by *vertex isoperimetric inequalities.* Such inequalities are particularly studied when $\mu(S)$ is sufficiently small (typically at most 1/2). These relationships are captured by the *isoperimetric parameter* (or *isoperimetric profile*) of a graph

$$\Phi(G, \nu) = \inf\{\mu(\partial S) \,:\, \mu(S) \geq \nu\}.$$

Proving such inequalities for various graphs is a frequent topic in the literature (e.g., [4, 8]). Typically such works focus on a *linear* or near-linear relationship between $\mu(\partial S)$ and $\mu(S)$, known as the *isoperimetric constant.*

$$h(G) = \inf\left\{\frac{\mu(\partial S)}{\mu(S)} \,\middle|\, S \subset V_G, \mu(S) \in (0, 1/2]\right\}. \tag{1}$$

In this paper, we study graphs for which there is an order-of-magnitude difference between $\mu(S)$ and $\mu(\partial S)$, when $\mu(S)$ is sufficiently small. For example, if $\mu(\partial S) \geq \sqrt{\mu(S)}$ for all S, we would like to say that G expands by a power of 2. Such 'hyper-expansion' can be captured by what we coin as the *isoperimetric exponent.* For all $\epsilon > 0$ consider.

$$\eta(G, \epsilon) = \inf\left\{\frac{\log \mu(S)}{\log \mu(\partial S)} \,\middle|\, S \subset V_G, \mu(S) \in (0, \epsilon)\right\} \tag{2}$$

where log is the natural logarithm. In other words, for every subset S of G of density δ, the boundary of S has density at least $\delta^{1/\eta(G,\epsilon)}$. The larger the parameter $\eta(G)$ is, the more 'expansive' the graph is. It is easy to see that $\eta(G, \epsilon)$ is in general a decreasing function of ϵ. As we often work with large subsets of our graph, we let $\eta(G) := \eta(G, 1)$.

In this paper, we study the isoperimetric profile of the tensor powers of cliques. For undirected graphs $G = (V_G, E_G), H = (V_H, E_H)$, we define the *tensor product* $G \otimes H$ to be the undirected graph on vertex set $V_1 \times V_2$ such that an edge connects (u_1, v_1) and (u_2, v_2) if and only if $\{u_1, u_2\} \in E_G$, and $\{v_1, v_2\} \in E_H$. Note that up to isomorphism, the tensor product is both commutative and associative. We then denote $\otimes^n G$ to be the tensor product of n copies of G. Since this is the only graph product discussed in this article, we shorten this to G^n. In this article, we focus on the case that $G = K_t$, where K_t is the complete graph on $t \geq 3$ vertices. It turns out for such graphs that for all $\epsilon > \frac{1}{t^n}$, $\eta(G) = \eta(G, \epsilon)$.

In particular, we shall compute the following.

▶ **Theorem 1.1.** *For all $t \geq 3$ and all positive integers n,*

$$\eta(K_t^n) = \eta(K_t) = \frac{\log t}{\log t - \log(t-1)} = t \log t + \Theta(\log t). \tag{3}$$

In addition to this high-level structure, we give a more-fine-tuned analysis of the behavior of $\Phi_t(\eta) := \inf_{n \geq 1} \Phi(K_t^n, \eta)$. (See Theorem 2.8.)

1.2 Independent set stability

Next, we apply these vertex isoperimetric inequalities to understand the structure of near-maximum independent sets of graphs. Such results are known as *stability* results.

Such results are not just of interest within combinatorics, a better understanding of independent set stability of certain graphs, such as K_t^n, have resulted in advances in hardness of approximation, particularly in construct dictatorship tests for approximate graph coloring and independent set problems (e.g., [1, 10, 7]). In fact the investigation which led to the results in this paper was inspired by the pursuit of such results.

A landmark result of this form due to [1] is as follows.

▶ **Theorem 1.2** ([1]). *For all $t \geq 3$ there exist C_t with the following property. For any positive integer n, Let $I \subset K_t^n$ be an independent set such that $\epsilon = 1 - t\mu(I)$, then there exists an independent set $J \subset K_t^n$ of maximum size ($\mu(J) = 1/t$) such that $\mu(I \Delta J) \leq C_t\epsilon$, where $S\Delta T = (S \setminus T) \cup (T \setminus S)$.*

In other words, independent sets of near-maximum size are similar in structure to the maximum independent sets. Note that if J is an independent set of maximum size, then for some $i \in [n]$ and $j \in [t]$, we have that

$$J = [t]^{i-1} \times \{j\} \times [t]^{n-i}.$$

This is a well-known result due to [22] (see [2] for a proof using Fourier analysis).

Ghandehari and Hatami improved this result (Theorem 1 of [21]) to show that if $t \geq 20$ and $\epsilon \leq 10^{-9}$ then C_t can be replaced with $40/t$. Both results were proven using Fourier analysis.

We improve upon this result in two steps. First, by applying Theorem 1.1 we improve Theorem 1.2 in a black-box matter to obtain the following result.

▶ **Theorem 1.3.** *For all $t \geq 3$, there exists $\epsilon_t > 0$ with the following property. For any positive integer n, let $I \subset K_t^n$ be an independent set such that $\epsilon = 1 - t\mu(I) < \epsilon_t$. Then there exists an independent set $J \subset K_t^n$ of maximum size ($\mu(J) = 1/t$) such that*

$$\mu(I \setminus J) \leq 4\epsilon^{\eta(K_t)} = 4\epsilon^{\log t/(\log t - \log(t-1))}. \tag{4}$$

▶ Remark 1.4. Since $\mu(I \setminus J) \leq 4\epsilon^{\eta(K_t)}$,

$$\mu(I\Delta J) = \mu(I \setminus J) + \mu(J \setminus I) = \mu(J) - \mu(I) + 2\mu(I \setminus J) = \frac{\epsilon}{t} + 8\epsilon^{\eta(K_t)},$$

so our result gives the optimal first-order structure for Theorem 1.2 assuming ϵ is sufficiently small. Furthermore, in Appendix B, we give examples of independent sets of K_t^n with arbitrarily small density (assuming $n \to \infty$) for which the exponent $\eta(K_t)$ is optimal.

Next, using a purely combinatorial argument we pin down a precise value for ϵ_t.

▶ **Theorem 1.5.** *In Theorem 1.3, for all $t \geq 3$, one may set $\epsilon_t = 1 - \frac{3}{t} + \frac{2}{t^2}$. In other words, the theorem applies for all independent sets I such that $\mu(I) > \frac{3t-2}{t^3}$.*

The choice of ϵ_t is not arbitrary, it corresponds to the density of the following independent set.

$$I = \{(1, 1, a), (1, a, 1), (a, 1, 1) \,:\, a \in [t]\} \times [t]^{n-3}.$$

Note that $\mu(I) = \frac{3t-2}{t^3}$. This set represents a phase transition in the independent sets from 'dictators' to 'juntas,' as the I constructed above is equally influenced by 3 coordinates (where 'influence' is in the sense of [1]). Such phase transitions have been studied in the literature [10], but this may be the first work to highlight the exact transition point.

Additionally, to the best of the author's knowledge, this is the first known purely combinatorial proof of Theorem 1.2.

1.3 Related work

Such stability results for independent sets have also been studied for Kneser graphs. A result similar to that of Theorem 1.2 was proved by [20]. Numerous other works in the literature [9, 11, 18, 19] use Fourier analysis to prove generalized stability results for Kneser graphs or other structures related to intersecting families. Other related works find purely combinatorial characterizations [3, 26, 27]. These results typically have a linear error bound ($\eta = 1$) on the closeness to maximal independent sets.

A result which also finds a "tight" super constant exponent $\eta > 1$ for the independent set stability is proved in some very recent work [14, 13, 16, 29, 28, 15] on Kneser graphs and related structures. (See also [12] and Proposition 4.3 of [17].) The techniques have high-level similarity to the ones adopted here:[1] particularly in their use of compressions to prove a isoperimetric inequality which they then bootstrap to a combinatorial independent set stability result.

1.4 Paper organization

In Section 2 we prove the claimed vertex isoperimetric inequalities. In Section 3, we prove the stability results for near-maximum independent sets in K_t^n. Appendix A proves some algebraic inequalities omitted from the main text. Appendix B shows that the exponent of $\eta(t)$ in Theorems 1.3 and 1.5 is optimal.

2 Vertex isoperimetric Inequalities

In this section, we proceed to prove the isoperimetry results claimed in Section 1.1.

Identify the vertex set of K_t^n with $[t]^n$. Two vertices of $x, y \in [t]^n$ are connected in K_t^n if and only if $x_i \neq y_i$ for all $i \in [n]$. Denote $y_{\neg i} := (y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n)$. We often write y as $(y_i, y_{\neg i})$ when it is clear from context which coordinate is being inserted.

2.1 Compressions

A useful tool in our study will be the operation of the well-known technique of *compressions* (e.g., [30, 31]). Although compressions are not strictly necessary to prove Theorem 1.1, they are essential in the proof of stronger isoperimetry results as well as Theorem 1.5, so we introduce the machinery now.

For $S \subseteq [t]^n$ be a subset, define the *compression of S in coordinate i* to be

$$c_i(S) = \{x \in [t]^n \,:\, x_i \leq |\{y \in S \,:\, y_{\neg i} = x_{\neg i}\}|\}\,. \tag{5}$$

This notion of compression appeared in the work of Bollobás and Leader [5].

Informally, we 'shift' each element of S to be as small as possible in the ith direction. Note that $\mu(c_i(S)) = \mu(S)$ for all $S \subseteq [t]^n$. It is easy to see that c_i is *idempotent*: $c_i(c_i(S)) = c_i(S)$ for all $S \subseteq [t]^n$ and $i \in [n]$.

We say that a set S is *compressed* if $c_i(S) = S$ for all $i \in [n]$. Equivalently, for all $x \in S$ there is no $y \in [t]^n \setminus S$ such that $x_i \leq y_i$ for all $i \in [n]$.

▶ Remark 2.1. Note that every time a compression c_i is applied, the quantity

$$\Sigma(S) := \sum_{x \in S} \sum_{j \in [n]} x_j$$

[1] The author became aware of these similar proofs only after writing major portions of the manuscript.

decreases or stays the same (in which case $c_i(S) = S$). Thus, since $\Sigma(S)$ is always positive, there must exist a finite sequence of compressions which can be applied to S to make the set compressed.

Now we show that compressions respect independent sets of K_t^n. This result is not needed until Section 3, but the proof does give intuition for how the compressions work.

▶ **Claim 2.2.** *For all $i \in [n]$ and all $I \subset [t]^n$ independent set of K_t^n, $c_i(I)$ is also an independent set of K_t^n.*

Proof. Assume not, then there exist $x, y \in c_i(I)$ such that $\{x, y\}$ is an edge. In particular, since $x_i \neq y_i$, we must have that $x_i \neq 1$ or $y_i \neq 1$. Assume without loss of generality that $y_i \neq 1$. Then, by definition of $c_i(I)$, there must be $z := (1, y_{\neg i}) \in c_i(I)$. Since $x, y, z \in c_i(I)$, there must be $x', y', z' \in I$ such that

$$\begin{aligned} x_{\neg i} &= x'_{\neg i} \\ y_{\neg i} = z_{\neg i} &= y'_{\neg i} = z'_{\neg i} \\ y'_i &\neq z'_i. \end{aligned}$$

Since $y'_i \neq z'_i$, we must either have that $x'_i \neq y'_i$ or $x'_i \neq z'_i$. In the former case, $\{x', y'\}$ is an edge of K_t^n and in the latter case $\{x', z'\}$ is an edge of K_t^n. This contradicts the fact that I is an independent set. ◀

Next we show that compressions can only decrease the size of the vertex boundary.

▶ **Claim 2.3.** *For all $i \in [n]$ and $S \subseteq [t]^n$, $|\partial c_i(S)| \leq |\partial S|$.*

Proof. Fix $\bar{a} := a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in [t]$. Consider $T = \{(a_1, \ldots, a_{i-1})\} \times [t] \times \{(a_{i+1}, \ldots, a_n\} \subset [t]^n$.

Note that for every vertex $v \in [t]^n$, $\partial\{v\} \cap T$ either has 0 or $t - 1$ elements. Thus, $|T \cap \partial S| \in \{0, t-1, t\}$. We claim that $|T \cap \partial c_i(S)| \leq |T \cap \partial S|$ for all T.

- If $|T \cap \partial S| = 0$, then there are no edges between S and T and shifting the vertices of S in the ith coordinate cannot change that. Thus, $|T \cap \partial c_i(S)| = 0$.
- If $|T \cap \partial S| = t - 1$, then the set $\partial T \cap S$ must be constant in the ith coordinate. Thus, $c_i(\partial T \cap S) = \partial T \cap c_i(S)$ is constant in the ith coordinate, so $|T \cap \partial c_i(S)| = t - 1$.
- If $|T \cap \partial S| = t$, then trivially $|T \cap \partial c_i(S)| \leq t$.

Thus, summing $|T \cap \partial c_i(S))| \leq |T \cap \partial S|$ across all possible T, we have that $|\partial c_i(S)| \leq |\partial S|$. ◀

▶ Remark 2.4. The proof crucially uses the fact that ∂S can include elements of S. If we instead had defined the vertex boundary to be $\partial S \setminus S$, there is a simple counterexample. Consider $t = 3$ and $n = 2$ and $S = \{(1, 2), (1, 3), (2, 1), (3, 1)\}$. Then it is not hard to check that $|\partial S| = |\partial c_1(S)| = 8$, but $|\partial S \setminus S| = 4 < 5 = |\partial c_1(S) \setminus c_1(S)|$.

2.2 Proof of Theorem 1.1

Define

$$\eta(t) := \frac{\log t}{\log t - \log(t-1)} = t \log t + \Theta(\log t). \tag{6}$$

First, we show that $\eta(K_t^n) \leq \eta(t)$. In fact, we show a whole family of equality cases.

▶ **Claim 2.5.** *For all positive integers n and t such that $t \geq 3$, $\eta(K_t^n) \leq \eta(t)$.*

Proof. For all integers $k \in [n]$, consider $S = \{1\}^k \times [t]^{n-k}$. Then $\partial S = \{2, \dots, t\}^k \times [t]^{n-k}$. Thus,

$$\eta(K_t^n) \leq \frac{\log \mu(S)}{\log \mu(\partial S)} = \frac{\log t^{-k}}{\log((t-1)^k t^{-k})} = \frac{k \log \frac{1}{t}}{k \log \frac{t-1}{t}} = \eta(t). \qquad ◀$$

The lower-bound is more difficult, we first need the following inequality, proved in Appendix A.

▶ **Claim 2.6.** *Let $t \geq 2$ be a positive integer and let $x \geq y \geq 0$ be real numbers, then*

$$y^{1/\eta(t)} + (t-1)x^{1/\eta(t)} \geq (t-1)(x + (t-1)y)^{1/\eta(t)} \tag{7}$$

▶ **Lemma 2.7.** *For positive integers $n \geq 1$ and $t \geq 3$ and all $S \subseteq [t]^n$, we have that*

$$\mu(\partial(S)) \geq \mu(S)^{1/\eta(t)}. \tag{8}$$

Therefore $\eta(K_t^n) \geq \eta(t)$.

Proof. By Claim 2.3 and Remark 2.1, it suffices to consider the case that S is compressed. We now proceed by induction on n.

For our base case, $n = 1$, we must have that $S = \emptyset$ in which case (8) is trivial, or $S = [k]$ for some positive integer $k \leq t$. If $S = [1]$, then $\partial S = \{2, \dots, t\}$, in which case we have an equality case of (8) by the proof of Claim 2.5. Otherwise, if $k \geq 2$, then $\partial S = [t]$, so $\mu(\partial S) = 1$, so (8) holds.

For $n \geq 2$, assume by the induction hypothesis that (8) is true for all $S \subseteq \mathbb{Z}_t^m$ where $1 \leq m < n$. For all $i \in [t]$, let

$$S_i := \{x_{\neg n} \,:\, x \in S, x_n = i\} \tag{9}$$

$$(\partial S)_i := \{x_{\neg n} \,:\, x \in \partial S, x_n = i\}. \tag{10}$$

Since S is compressed for all $1 \leq i \leq j \leq t$, we have that $S_i \supseteq S_j$. Thus, if $i \in \{2, \dots, t\}$ is nonzero, for any $x \in (\partial S)_i$, there is $y \in S_1$ connected to x by an edge of K_t^{n-1}. Thus, $\partial S_1 \subseteq (\partial S)_i$. Similarly, for any $x \in (\partial S)_1$, there is $y \in S_2$ such that x is disjoint from y. Therefore, $\partial S_2 \subseteq (\partial S)_1$. Putting these together,

$$\begin{aligned}
\mu(\partial S) &= \frac{1}{t} \sum_{i \in [t]} \mu((\partial S)_i) \\
&\geq \frac{1}{t}(\mu(\partial S_2) + (t-1)\mu(\partial S_1)) \\
&\geq \frac{1}{t}\left(\mu(S_2)^{1/\eta(t)} + (t-1)\mu(S_1)^{1/\eta(t)}\right),
\end{aligned}$$

where we applied the inductive hypothesis in the last step. Applying Claim 2.6, using the fact that $0 \le \mu(S_2) \le \mu(S_1)$, we have that

$$\begin{aligned}
\mu(\partial(S)) &\ge \frac{1}{t}\left(\mu(S_2)^{1/\eta(t)} + (t-1)\mu(S_1)^{1/\eta(t)}\right)\\
&\ge \frac{t-1}{t}\left(\mu(S_1) + (t-1)\mu(S_2)\right)^{1/\eta(t)}\\
&\ge \frac{t-1}{t}\left(\sum_{i\in[t]} \mu(S_i)\right)^{1/\eta(t)}\\
&= \left(\frac{1}{t}\sum_{i\in[t]} \mu(S_i)\right)^{1/\eta(t)}\\
&= \mu(S)^{1/\eta(t)},
\end{aligned}$$

as desired. ◀

Claim 2.5 and Lemma 2.7 together imply Theorem 1.1.

2.3 A fine-tuned understanding of the isoperimetric profile

Recall that the (vertex) isoperimetric profile of a graph G is

$$\Phi(G,\nu) := \inf\{\mu(\partial S) \,:\, \mu(S) \ge \nu\}.$$

For $t \ge 3$ fixed, define

$$\Phi_t(\nu) := \inf_{n\ge 1} \Phi(K_t^n, \nu).$$

Note that Φ_t is non-decreasing. To avoid complications with the discrete behavior of $\Phi(K_t^n,\nu)$ when n is small, it is easier to instead work with $\Phi_t(\nu)$. By Theorem 1.1,

$$\Phi_t(\nu) \ge \nu^{1/\eta(t)}. \tag{11}$$

This is tight whenever $\nu = t^{-k}$ for any integer $k \ge 0$, but ceases to be tight when $\log_t(\nu)$ is non-integral (see Figure 1).

The following recursive relationship allows one to compute $\Phi_t(\nu)$ to arbitrary precision.

▶ **Theorem 2.8.** *For all $t \ge 3$,*

$$\Phi_t(\nu) = \begin{cases} \frac{t-1}{t}\Phi_t(t\nu) & \nu < 1/t \\ \frac{t-1}{t} + \frac{1}{t}\Phi_t\left(\frac{t\nu-1}{t-1}\right) & \nu \ge 1/t \end{cases}. \tag{12}$$

Using the simple fact that $\Phi_t(0) = 0$ and $\Phi_t(1) = 1$, the above equation is extremely powerful. For example,

$$\Phi_3\left(\frac{5}{9}\right) = \frac{2}{3} + \frac{1}{3}\Phi_3\left(\frac{1}{3}\right) = \frac{8}{9},$$

which is an exact bound compared to $(\frac{5}{9})^{1/\eta(3)} \approx \frac{7.24}{9}$. This recursion is what allowed the creation of Figure 1.

Theorem 2.8 is proved in the full version. This more refined understanding of Φ_t proves critical in the combinatorial proof of Theorem 1.5.

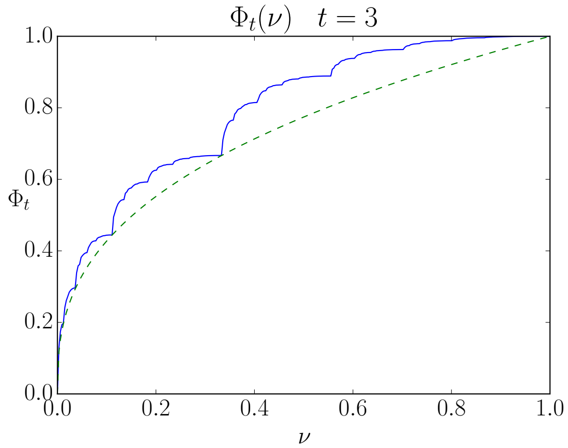


Figure 1 A graph of $\Phi_t(\nu)$ for $t = 3$. The dashed curve $\nu^{1/\eta(t)}$ is for reference.

3 Independent set stability results

In this section, we seek to prove the main independent set stability result, Theorem 1.5. This is done in two stages. First, we prove a simpler version (Theorem 1.3) where we use the weaker vertex isoperimetry inequality to amplify Theorem 1.2 of [1] in a "black-box" manner. Second, we utilize the fine-grained vertex isoperimetry inequality in a fully combinatorial inductive proof to obtain the full Theorem 1.5 without dependence of Theorem 1.2 of [1].

3.1 Black-box result for clique tensor powers

First, we show that if a large independent set I is somewhat close to a maximum-sized independent set J, then it is really close to J. We fix positive integers n and $t \geq 3$.

▶ **Lemma 3.1.** *Let $I \subset [t]^n$ be an independent set with $\epsilon := 1 - t\mu(I)$. Assume there exists a maximum-sized independent set J such that*

$$\mu(I \setminus J) < \frac{1}{t^3}.$$

Then,

$$\mu(I \setminus J) < 4\epsilon^{\eta(t)}.$$

Proof. Without loss of generality, we may assume that $J = [t]^{n-1} \times [1]$. Pick $J' = [t]^{n-1} \times \{j\}$ such that $j \neq 1$ but otherwise $\mu(I \cap J')$ is maximal. Let $\delta := \mu(I \setminus J)$. Since J and J' are disjoint, we have that

$$\mu(I \cap J') \geq \frac{\mu(I \setminus J)}{t-1} = \frac{\delta}{t-1}.$$

Now, consider $S = \partial(I \cap J')$. Recall the definition of $S_k \subseteq [t]^{n-1}$ from (9). Since $I \cap J' \subseteq J'$ has the property that every element has the same last coordinate, $S_k = S_{k'}$ for all $k, k' \neq j$ and $S_j = \emptyset$. Thus, $\mu(S_k) = \frac{t}{t-1}\mu(S)$ for all $k \neq j$. Therefore,

$$\mu(S \cap J) = \frac{1}{t}\mu((S \cap J)_i) = \frac{1}{t}\mu(S_i) = \frac{1}{t-1}\mu(S).$$

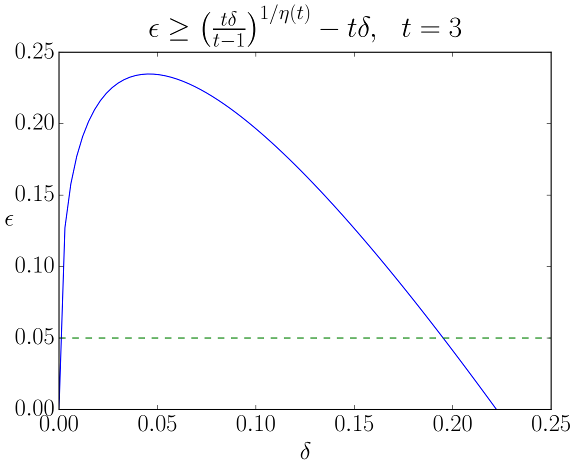


Figure 2 Plot of (15) when $t = 3$. Notice the bifurcation of solutions to (15) for a fixed ϵ (line $\epsilon = 0.05$ is dashed).

Applying Theorem 1.1, we get that

$$\mu(S \cap J) = \frac{1}{t-1}\mu(\partial(I \cap J')) \geq \frac{1}{t-1}\mu(I \cap J')^{1/\eta(t)} \geq \frac{1}{t-1}\left(\frac{\delta}{t-1}\right)^{1/\eta(t)}.$$

Since I is an independent set, ∂I is disjoint from I. Since $S \cap J = \partial(I \cap J') \cap J \subseteq \partial I$, we have that $I \cap J$ and $S \cap J$ are disjoint. Therefore,

$$\mu(I \cap J) \leq \mu(J) - \mu(S \cap J) \leq \frac{1}{t} - \frac{1}{t-1}\left(\frac{\delta}{t-1}\right)^{1/\eta(t)}. \tag{13}$$

But, we also know that

$$\mu(I \cap J) = \mu(I) - \mu(I \setminus J) = \frac{1}{t}(1-\epsilon) - \delta. \tag{14}$$

By (13) and (14)

$$\frac{1}{t}(1-\epsilon) - \delta \leq \frac{1}{t} - \frac{1}{t-1}\left(\frac{\delta}{t-1}\right)^{1/\eta(t)} = \frac{1}{t} - \frac{1}{t}\left(\frac{t\delta}{t-1}\right)^{1/\eta(t)}.$$

Thus,

$$\epsilon \geq \left(\frac{t\delta}{t-1}\right)^{1/\eta(t)} - t\delta \geq \delta^{1/\eta(t)} - t\delta. \tag{15}$$

Consider Figure 2 which has a plot of the RHS of (15) when $t = 3$. If ϵ is sufficiently small, then the inequality holds only when δ is very small (polynomial in ϵ) or very large (about $\frac{1}{t}$). Since is 'moderately' small ($\delta \leq \frac{1}{t^3}$), we must have that δ is very small. Quantitatively, note that

$$\begin{aligned} t\delta &= t\delta^{1/\eta(t)}\delta^{1-1/\eta(t)} \\ &\leq t\delta^{1/\eta(t)}\left(\frac{1}{t^3}\right)^{1-1/\eta(t)} \\ &= t\delta^{1/\eta(t)}\frac{1}{t^3}\left(\frac{t^3}{(t-1)^3}\right) \\ &\leq \frac{t\delta^{1/\eta(t)}}{(t-1)^3}. \end{aligned}$$

So

$$\epsilon \geq \delta^{1/\eta(t)}\left(1 - \frac{t}{(t-1)^3}\right).$$

Therefore,

$$\delta \leq \left(\frac{(t-1)^3}{(t-1)^3 - t}\right)^{\eta(t)} \epsilon^{\eta(t)} \leq 4\epsilon^{\eta(t)},$$

where the last inequality follows from the following claim which is proved in Appendix A.

▶ **Claim 3.2.** *For all $t \geq 3$,*

$$\left(\frac{(t-1)^3}{(t-1)^3 - t}\right)^{\eta(t)} \leq 4.$$ ◀

We now use this lemma to 'amplify' Theorem 1.2 to prove Theorem 1.3.

Proof of Theorem 1.3. Set $\epsilon_t := \frac{1}{C_t t^3} > 0$. Consider any independent set I of of K_t^n such that $\epsilon := 1 - t\mu(I) < \epsilon_t$. Pick any maximum-sized J guaranteed by Theorem 1.2 such that

$$\delta := \mu(I \setminus J) \leq \mu(I \Delta J) \leq C_t \epsilon < \frac{1}{t^3}. \tag{16}$$

By Lemma 3.1, we have that

$$\delta \leq 4\epsilon^{\eta(t)},$$

as desired. ◀

3.2 Improved stability result for clique tensor powers

In this section we improve ϵ_t in Theorem 1.3 to an explicit expression. In fact, we may show that

$$\epsilon_t = 1 - \frac{3}{t} + \frac{2}{t^2}$$

which corresponds to independent sets I for which $\mu(I) > \frac{3t-2}{t^3}$.

Proofs of claims and lemmas in this section are reserved for the full version.

First, we try to show that if an independent set I is large enough, then I is either very close to or very far from a maximum-sized independent set. To do this, we show that if I is 'moderately far' from a maximum-sized independent set, then this moderate-sized portion which is not in the maximum-sized independent set has such a large vertex boundary that it precludes a large portion of the maximum-sized independent set from being part of I, forcing the density of I to be at or below our threshold of $\frac{3t-2}{t^3}$.

We need a notation for the maximum sized independent sets. For all $i \in [t]$ and $j \in [n]$ let

$$J_{i,j} = [t]^{j-1} \times \{i\} \times [t]^{n-j}. \tag{17}$$

We say that I is *sorted* if there exists that for all $i_1, i_2 \in [t]$ and $j \in [n]$ we have that $i_1 \leq i_2$ implies that

$$\mu(I \cap J_{i_1,j}) \leq \mu(I \cap J_{i_2,j}).$$

Note that unlike compressions, we may assume without loss of generality that I is sorted since permuting the labels so that an independent set is sorted does not change its intersection sizes with the maximum independent sets.

▶ **Claim 3.3.** *Let $I \subset [t]^n$ be a sorted independent set such that $\mu(I) > \frac{3t-2}{t^3}$ (or $1-t\mu(I) < \epsilon_t$), then for all $j \in [n]$,*

$$\mu(I \setminus J_{1,j}) < \frac{t-1}{t^4} \text{ or } \mu(I \setminus J_{1,j}) > \frac{t-1}{t^3}. \tag{18}$$

From Theorem 2.8, we can attain a bound that is even better.

▶ **Claim 3.4.** *Let $I \subset [t]^n$ be a sorted independent set such that $\mu(I) > \frac{3t-2}{t^3}$, then for all $j \in [n]$,*

$$\mu(I \setminus J_{1,j}) < \frac{t-1}{t^4} \text{ or } \mu(I \setminus J_{1,j}) > \frac{(2t-1)(t-1)}{t^4}. \tag{19}$$

The next key step is to show Theorem 1.5 essentially holds for *compressed* independent sets I.

▶ **Lemma 3.5.** *Let $I \subset [t]^n$ be a compressed independent set such that $\mu(I) > \frac{3t-2}{t^3}$, then for some $j \in [n]$,*

$$\mu(I \setminus J_{1,j}) < \frac{t-1}{t^4}. \tag{20}$$

Note that by Lemma 3.1, we immediately have that Theorem 1.5 holds for compressed independent sets.

Now we extend this result to sorted independent sets; and thus all independent sets.

▶ **Lemma 3.6.** *Let $I \subset [t]^n$ be a sorted independent set such that $\mu(I) > \frac{3t-2}{t^3}$, then for some $j \in [n]$,*

$$\mu(I \setminus J_{1,j}) < \frac{t-1}{t^4}. \tag{21}$$

Proof of Theorem 1.5. Let $I \subset [t]^n$ be an independent set with $\mu(I) > \frac{3t-2}{t^3}$. Assume without loss of generality that I is sorted. By Lemma 3.6, we know that there is $j \in [n]$ such that

$$\mu(I \setminus J_{1,j}) \le \frac{t-1}{t^4} < \frac{1}{t^3}.$$

Thus, by Lemma 3.1, we have that

$$\mu(I \setminus J_{1,j}) \le 4\epsilon^{\eta(t)},$$

as desired. ◀

Acknowledgments. The author is indebted to Venkatesan Guruswami for numerous insightful discussions and comments, in particular for pointing the author to [1].

The author would also like to thank Boris Bukh and Po-Shen Loh for helpful comments and discussions as well as David Ellis and anonymous reviewers for comments on an earlier version of the manuscript.

The 2D plots were created using Matplotlib [25]. The 3D visualizations were created using Asymptote [23].

References

1 N. Alon, I. Dinur, E. Friedgut, and B. Sudakov. Graph Products, Fourier Analysis and Spectral Techniques. *Geometric & Functional Analysis GAFA*, 14(5):913–940, 2004. doi:10.1007/s00039-004-0478-3.

2 Noga Alon and Joel H. Spencer. *The Probabilistic Method*. John Wiley & Sons, April 2004. Google-Books-ID: q3lUjheWiMoC.

3 József Balogh and Dhruv Mubayi. A new short proof of a theorem of Ahlswede and Khachatrian. *Journal of Combinatorial Theory, Series A*, 115(2):326–330, February 2008. doi:10.1016/j.jcta.2007.03.010.

4 S. Bobkov, C. Houdré, and P. Tetali. λ_∞, Vertex Isoperimetry and Concentration. *Combinatorica*, 20(2):153–172, February 2000. doi:10.1007/s004930070018.

5 Béla Bollobás and Imre Leader. Compressions and isoperimetric inequalities. *Journal of Combinatorial Theory, Series A*, 56(1):47–62, January 1991. doi:10.1016/0097-3165(91)90021-8.

6 Joshua Brakensiek. Vertex isoperimetry and independent set stability for tensor powers of cliques. *arXiv:1702.04432 [cs, math]*, February 2017. arXiv: 1702.04432. URL: http://arxiv.org/abs/1702.04432.

7 Joshua Brakensiek and Venkatesan Guruswami. New Hardness Results for Graph and Hypergraph Colorings. In Ran Raz, editor, *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 14:1–14:27, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2016.14.

8 Demetres Christofides, David Ellis, and Peter Keevash. An Approximate Vertex-Isoperimetric Inequality for r-sets. *The Electronic Journal of Combinatorics*, 20(4):P15, August 2013. URL: http://www.combinatorics.org/ojs/index.php/eljc/article/view/v20i4p15.

9 Irit Dinur and Ehud Friedgut. Intersecting Families Are Essentially Contained in Juntas. *Comb. Probab. Comput.*, 18(1-2):107–122, March 2009. doi:10.1017/S0963548308009309.

10 Irit Dinur, Ehud Friedgut, and Oded Regev. Independent Sets in Graph Powers are Almost Contained in Juntas. *Geometric and Functional Analysis*, 18(1):77–97, April 2008. doi:10.1007/s00039-008-0651-1.

11 Irit Dinur and Samuel Safra. On the Hardness of Approximating Minimum Vertex Cover. *Annals of Mathematics*, 162(1):439–485, 2005. URL: http://www.jstor.org/stable/3597377.

12 David Ellis, Gil Kalai, and Bhargav Narayanan. On symmetric intersecting families. *arXiv:1702.02607 [math]*, February 2017. arXiv: 1702.02607. URL: http://arxiv.org/abs/1702.02607.

13 David Ellis, Nathan Keller, and Noam Lifshitz. On the structure of subsets of the discrete cube with small edge boundary. *arXiv:1612.06680 [math]*, December 2016. arXiv: 1612.06680. URL: http://arxiv.org/abs/1612.06680.

14 David Ellis, Nathan Keller, and Noam Lifshitz. Stability versions of Erdős-Ko-Rado type theorems, via isoperimetry. *arXiv:1604.02160 [math]*, April 2016. arXiv: 1604.02160. URL: http://arxiv.org/abs/1604.02160.

15 David Ellis, Nathan Keller, and Noam Lifshitz. On a Biased Edge Isoperimetric Inequality for the Discrete Cube. *arXiv:1702.01675 [math]*, February 2017. arXiv: 1702.01675. URL: http://arxiv.org/abs/1702.01675.

16 David Ellis and Noam Lifshitz. On the union of intersecting families. *arXiv:1610.03027 [math]*, October 2016. arXiv: 1610.03027. URL: http://arxiv.org/abs/1610.03027.

17 Yuval Filmus. Ahlswede-Khachatrian Theorems: Weighted, Infinite, and Hamming. *arXiv:1610.00756 [math]*, October 2016. arXiv: 1610.00756. URL: `http://arxiv.org/abs/1610.00756`.

18 Yuval Filmus, Guy Kindler, Elchanan Mossel, and Karl Wimmer. Invariance Principle on the Slice. In Ran Raz, editor, *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:10, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.CCC.2016.15`.

19 Yuval Filmus and Elchanan Mossel. Harmonicity and Invariance on Slices of the Boolean Cube. In Ran Raz, editor, *31st Conference on Computational Complexity (CCC 2016)*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:13, Dagstuhl, Germany, 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.CCC.2016.16`.

20 Ehud Friedgut. On the measure of intersecting families, uniqueness and stability. *Combinatorica*, 28(5):503–528, September 2008. `doi:10.1007/s00493-008-2318-9`.

21 Mahya Ghandehari and Hamed Hatami. Fourier analysis and large independent sets in powers of complete graphs. *Journal of Combinatorial Theory, Series B*, 98(1):164–172, January 2008. `doi:10.1016/j.jctb.2007.06.003`.

22 D. Greenwell and L. Lovász. Applications of product colouring. *Acta Mathematica Academiae Scientiarum Hungarica*, 25(3-4):335–340, September 1974. `doi:10.1007/BF01886093`.

23 Andy Hammerlindl, John Bowman, and Tom Prince. Asymptote: The vector graphics language, 2014.

24 A. J. W. Hilton and E. C. Milner. Some intersection theorems for systems of finite sets. *The Quarterly Journal of Mathematics*, 18:369–384, 1967. `doi:10.1093/qmath/18.1.369`.

25 John D. Hunter. Matplotlib: A 2d Graphics Environment. *Computing in Science & Engineering*, 9(3):90–95, May 2007. `doi:10.1109/MCSE.2007.55`.

26 Peter Keevash. Shadows and intersections: Stability and new proofs. *Advances in Mathematics*, 218(5):1685–1703, August 2008. `doi:10.1016/j.aim.2008.03.023`.

27 Peter Keevash and Dhruv Mubayi. Set systems without a simplex or a cluster. *Combinatorica*, 30(2):175–200, March 2010. `doi:10.1007/s00493-010-2401-x`.

28 Nathan Keller and Noam Lifshitz. On Large H-Intersecting Families. *arXiv:1609.01884 [math]*, September 2016. arXiv: 1609.01884. URL: `http://arxiv.org/abs/1609.01884`.

29 Nathan Keller and Noam Lifshitz. A tight stability version of the Complete Intersection Theorem. *arXiv:1604.06135 [math]*, April 2016. arXiv: 1604.06135. URL: `http://arxiv.org/abs/1604.06135`.

30 N. Sauer. On the density of families of sets. *Journal of Combinatorial Theory, Series A*, 13(1):145–147, July 1972. `doi:10.1016/0097-3165(72)90019-2`.

31 Saharon Shelah. A combinatorial problem; stability and order for models and theories in infinitary languages. *Pacific Journal of Mathematics*, 41(1):247–261, April 1972. URL: `http://msp.org/pjm/1972/41-1/p21.xhtml`.

A Proofs of algebraic inequalities

Proof of Claim 2.6. Let $\alpha(t) = 1/\eta(t)$. For $c \ge 0$, let $f_c(z) = (z+c)^{\alpha(t)} - z^{\alpha(t)}$. Notice that if $z > 0$, then $f_c'(z) = (\alpha(t))((z+c)^{\alpha(t)-1} - z^{\alpha(t)-1}) \le 0$. Thus, we have that $(t-1)f_c(y) \ge (t-1)f_c(x)$ for all $c \ge 0$. Consider $c = (t-1)y$; we then have that

$$(t-1)f_c(y) = (t-1)((ty)^{\alpha(t)} - y^{\alpha(t)}) = (t-1)(t^{\alpha(t)} - 1)y^{\alpha(t)} = y^{\alpha(t)} \ge$$
$$(t-1)f_c(x) = (t-1)((x+(t-1)y)^{\alpha(t)} - x^{\alpha(t)}).$$

Rearranging, we obtain (7). ◀

Proof of Claim 3.2. First, verify the cases $t = 3$ and $t = 4$ using a calculator. Notice that $\eta(t) = \frac{\log t}{\log t - \log(t-1)} \leq t \log t$ so

$$\left(\frac{(t-1)^3}{(t-1)^3 - t}\right)^{\eta(t)} \leq e^{\frac{t\eta(t)}{(t-1)^3-t}} \leq e^{\frac{t^2 \log t}{(t-1)^3-t}}.$$

Also use a calculator to verify that $h(t) := \frac{t^2 \log t}{(t-1)^3-t}$ is less than 1 for $t = 5$. Now observe that when going from t to $t+1$, the numerator increases by

$$\begin{aligned}(t+1)^2 \log(t+1) - t^2 \log t &= (2t+1)\log(t+1) + t^2 \log(1 + \frac{1}{t}) \\ &\leq (2t+1)\log(t+1) + t \leq (2t+1)t + t \\ &= 2t^2 + 2t.\end{aligned}$$

and the denominator increases by

$$t^3 - (t+1) - (t-1)^3 + t = 3t^2 - 3t$$

Since $2t^2 + 2t \leq 3t^2 - 3t$ for all $t \geq 5$ and $h(5) \leq 1$, we have by a simple inductive proof that $h(t) \leq 1$ for all $t \geq 5$. Thus, for all $t \geq 5$,

$$\left(\frac{(t-1)^3}{(t-1)^3 - t}\right)^{\eta(t)} \leq e^1 < 4,$$

as desired. ◀

B Optimality of exponent in Theorem 1.3

In this appendix, we show in (4) of Theorem 1.3 that the exponent $\eta(t) = \frac{\log t}{\log t - \log(t-1)}$ is optimal and that the constant factor of 4 is nearly optimal. In other words, the stability result is optimal up to a constant factor.

▶ **Lemma 2.1.** *For all $t \geq 3$, there exists an infinite sequence of independent sets $\{I_n\}_{n \geq 3}$ such that $I_n \subset [t]^n$, $\epsilon_n = 1 - t\mu(I_n) > 0$ tends to 0 as $n \to \infty$, and for any n and any maximum-sized independent set J_n of K_t^n,*

$$\mu(I_n \setminus J_n) > \frac{t-1}{t}\epsilon^{\eta(t)}.$$

Proof. For $n \geq 3$, consider $J_n = [1] \times [t]^{n-1}$ and

$$I_n := (([t] \times [1]^{n-1}) \cup J_n) \setminus ([1] \times \{2, \ldots, t-1\}^n) \tag{22}$$

See Figure 3 for a visualization. It has been noted to the author that this construction is similar in structure to the constructions in the Hilton-Milner theorem [24].

One may check that I_n is an independent set of K_t^n and J_n is a maximum-sized independent set which minimizes $\mu(I_n \setminus J_n)$. Furthermore,

$$\mu(I_n) = \frac{t-1}{t^n} + \frac{1}{t} - \frac{(t-1)^{n-1}}{t^n}.$$

Thus,

$$\epsilon_n = \frac{(t-1)^{n-1} - (t-1)}{t^{n-1}} \tag{23}$$

$$\delta_n := \mu(I_n \setminus J_n) = \frac{t-1}{t^n}. \tag{24}$$

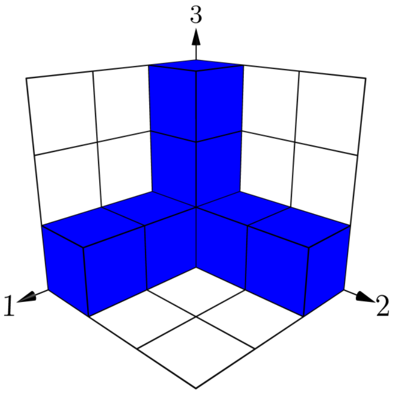


Figure 3 Schematic of I_3 when $t = 3$.

Notice that since $t^{1/\eta(t)} = \frac{t-1}{t}$.

$$\begin{aligned}\delta_n^{1/\eta(t)} &= \frac{(t-1)^{1/\eta(t)}}{t^{n/\eta(t)}}\\ &= \left(\frac{t-1}{t}\right)^{1/\eta(t)} \left(\frac{t-1}{t}\right)^{n-1}\\ &= \left(\frac{t-1}{t}\right)^{1/\eta(t)} (\epsilon_n + t\delta_n)\\ &> \left(\frac{t-1}{t}\right)^{1/\eta(t)} \epsilon_n.\end{aligned}$$

Therefore, raising both sides to the $\eta(t)$ power,

$$\delta_n > \frac{t-1}{t}\epsilon_n^{\eta(t)},$$

as desired. ◀

Polynomial Mixing of the Edge-Flip Markov Chain for Unbiased Dyadic Tilings*

Sarah Cannon[1], David A. Levin[2], and Alexandre Stauffer[3]

1 **College of Computing, Georgia Institute of Technology, Atlanta, GA, USA**
sarah.cannon@gatech.edu

2 **Department of Mathematics, University of Oregon, Eugene, OR, USA**
dlevin@uoregon.edu

3 **Department of Mathematical Sciences, University of Bath, Bath, UK**
a.stauffer@bath.ac.uk

Abstract

We give the first polynomial upper bound on the mixing time of the edge-flip Markov chain for unbiased dyadic tilings, resolving an open problem originally posed by Janson, Randall, and Spencer in 2002 [16]. A *dyadic tiling* of size n is a tiling of the unit square by n non-overlapping dyadic rectangles, each of area $1/n$, where a *dyadic rectangle* is any rectangle that can be written in the form $[a2^{-s}, (a+1)2^{-s}] \times [b2^{-t}, (b+1)2^{-t}]$ for $a, b, s, t \in \mathbb{Z}_{\geq 0}$. The edge-flip Markov chain selects a random edge of the tiling and replaces it with its perpendicular bisector if doing so yields a valid dyadic tiling. Specifically, we show that the relaxation time of the edge-flip Markov chain for dyadic tilings is at most $O(n^{4.09})$, which implies that the mixing time is at most $O(n^{5.09})$. We complement this by showing that the relaxation time is at least $\Omega(n^{1.38})$, improving upon the previously best lower bound of $\Omega(n \log n)$ coming from the diameter of the chain.

1998 ACM Subject Classification G.3 Markov Processes, G.2.1 Combinatorics

Keywords and phrases Random dyadic tilings, spectral gap, rapid mixing

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.34

1 Introduction

We study the edge-flip Markov chain for dyadic tilings. An interval is *dyadic* if it can be written in the form $[a2^{-s}, (a+1)2^{-s}]$ for non-negative integers a and s with $0 \leq a < 2^s$. A rectangle is dyadic if it is the Cartesian product of two dyadic intervals. A *dyadic tiling of size n* is a tiling of the unit square by n non-overlapping dyadic rectangles with the same area $1/n$; see Figure 1. Lagarias, Spencer, and Vinson [17] showed that dyadic tilings are precisely those tilings that can be constructed by bisecting the unit square, either horizontally or vertically; bisecting each half again, either horizontally or vertically; and repeatedly bisecting all remaining rectangular regions until there are n total dyadic rectangles, each of equal area. We necessarily assume n is a power of 2. There is a natural Markov chain which connects the state space of all dyadic tilings of size n by moves we refer to as *edge-flips*.

We analyze this *edge-flip Markov chain* over the set of dyadic tilings of size n. Given any dyadic tiling, this chain evolves by selecting an edge of the tiling uniformly at random and replacing it by its perpendicular bisector, if doing so yields a valid dyadic tiling of size

* This work was partially supported by NSF DGE-1650044 and a grant from the Simons Foundation (#361047) (Cannon), a Marie Curie Career Integration Grant PCIG13-GA-2013-618588 DSRELIS and an EPSRC Early Career Fellowship (Stauffer).

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 34; pp. 34:1–34:21
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

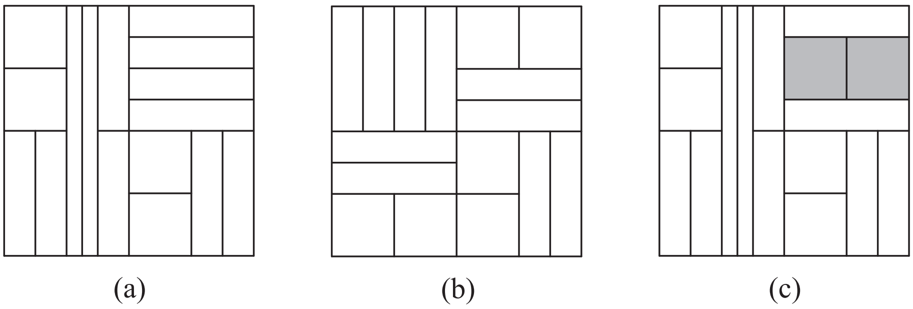


Figure 1 (a) A dyadic tiling of size 16 with a vertical bisector. (b) A dyadic tiling of size 16 with both a vertical and horizontal bisector. (c) A tiling that is not dyadic; the vertical component of the shaded rectangles is not a dyadic interval.

n; an illustration is given in Figure 2(a). Our main result gives the first polynomial upper bound for the mixing time of this Markov chain. (The precise definitions of mixing time and relaxation time are deferred to Section 2.2.) *In this paper, all logarithms have base 2.*

▶ **Theorem 1.** *The relaxation time of the edge-flip Markov chain for dyadic tilings of size n is at most $O(n^{\log 17})$. As a consequence, the mixing time of this chain is at most $O(n^{1+\log 17})$.*

The best previously known lower bound for the mixing time is $\Omega(n \log n)$, which is a simple consequence of the fact that the diameter of the Markov chain is of order $n \log n$ [16]. In the theorem below we improve this bound.

▶ **Theorem 2.** *The relaxation time and mixing time of the edge-flip Markov chain for dyadic tilings of size n are both at least $\Omega(n^{2\log\phi})$, where $\phi = \frac{\sqrt{5}+1}{2}$ is the golden ratio.*

We note that $\log 17 \sim 4.09$ and $2\log\phi \sim 1.38$.

1.1 Related work

The edge-flip Markov chain for dyadic tilings was first considered by Janson, Randall, and Spencer in 2002 [16], who showed it is irreducible but left as an open problem to derive that the mixing time is polynomial in n. Instead, they presented another Markov chain, which has additional global moves consisting of rotations at all scales, and showed that this chain mixes in polynomial time. However, applications of the comparison technique of Diaconis and Saloff-Coste [10] have failed to extend this polynomial mixing bound to the more natural edge-flip Markov chain (which, in fact, corresponds to only performing rotations at the smallest scale).

Cannon, Miracle, and Randall considered the mixing time of the edge-flip Markov chain for a weighted version of dyadic tilings [3]. In this version, given a parameter $\lambda > 0$, the stationary probability of a dyadic tiling x is proportional to $\lambda^{|x|}$, where $|x|$ is the sum of the length of the edges of x. The Metropolis rule [24] is incorporated into the edge-flip Markov chain so that the chain has the desired stationary distribution. They showed the mixing time of this chain is at least exponential in n^2 for any $\lambda > 1$, and at most $O(n^2 \log n)$ for any $\lambda < 1$. This establishes a phase transition at critical point $\lambda = 1$, which corresponds to the unweighted case considered here. However, their techniques did not extend to the critical point, and they left as an open problem bounding the mixing time when $\lambda = 1$; it is notoriously often quite difficult to bound mixing times at or near critical points. Our main result, Theorem 1, uses a different, non-local approach to finally answer the question of [16] and [3] by showing the mixing time of the edge-flip Markov chain at critical point $\lambda = 1$ is at most polynomial in n, substantially less than the mixing time when $\lambda > 1$. Furthermore, our Theorem 2 combined with the result for the weighted case in [3] shows that the behavior at the (unweighted) critical point $\lambda = 1$ is also substantially different than when $\lambda < 1$. While it follows from the path coupling analysis in [3] that the relaxation time is $O(n)$ for all fixed $\lambda < 1$, Theorem 2 establishes a super-linear lower bound on the relaxation time when $\lambda = 1$. (The path coupling technique is due to [1].)

It is a general principle in statistical physics that in systems with some bias parameter (*temperature*) that induces different phases, the mixing time of the natural heat-bath dynamics should be as fast as possible at high temperature, a larger polynomial at the critical temperature, and exponential at low temperature. (See [20] for a precise statement for the Ising model on the square lattice.) However, there are very few instances for which this behavior has been rigorously confirmed. Exceptions are the Ising model on complete graphs [18, 11], regular trees [12], and the two-dimensional lattice [20], and the Potts model on the complete graph [9] and the two-dimensional lattice [13], all of which required significant effort to analyze. The edge-flip Markov chain for dyadic tilings is an example of heat-bath dynamics, and the parameter λ introduced by Cannon et al. can be viewed as a function of inverse temperature. Their work confirms exponential mixing at low temperature ($\lambda > 1$) and polynomial mixing at high temperature ($\lambda < 1$). Our work shows that the mixing time at the critical point ($\lambda = 1$) is indeed polynomial but strictly larger than the diameter of the state space (which is $n \log(n)/2$), providing further evidence for the general statistical physics principle above.

Variants of the edge-flip Markov chain offer a natural way to sample from many structures, but establishing rigorous polynomial upper bounds on the mixing time has often proven difficult, even in simple cases. Perhaps the most studied case is that of triangulations of a given point set; efficiently generating uniformly random triangulations of general planar point sets has been a problem of great interest in computer graphics and computational geometry. However, the mixing time of the edge-flip Markov chain for triangulations remains open in the general case, and no polynomial upper bound is known. The only known exception is for n points in *convex position*, which corresponds to triangulations of a convex polygon. In this case, the edge-flip Markov chain is known to mix in at most $O(n^5)$ steps [23], but the correct order of the mixing time is still unknown. For the case of lattice triangulations, which are triangulations of an $m \times n$ grid of points, no polynomial upper bound on the mixing time is known even when $m \geq 2$ is kept fixed as $n \to \infty$. The only known results in this case are limited to the weighted case [4, 5, 27].

Another example of a related Markov chain that uses natural edge-flip type moves is the *switch Markov chain* for sampling from graphs with a given degree sequence. In this chain, at each iteration two random non-adjacent edges are removed and their four endpoints are randomly rematched; the move is rejected if it results in a multiple edge. Again, in the general case the mixing time of this Markov chain is unknown, though polynomial upper bounds exist when certain restrictions are placed on the degree sequence [8, 14].

For the case of rectangular tilings, results for the mixing time of the edge-flip Markov chain have been quite rare. One important result was obtained for domino tilings, which are tilings of an $n \times n$ square by rectangles of dimensions 1×2 or 2×1. In this case, the edge-flip Markov chain is known to mix in time polynomial in the number of dominoes, a result that heavily relies on the connection between domino tilings and random lattice paths [21, 25].

The case of dyadic tilings exhibits interesting asymptotic properties that have been studied by combinatorialists [17, 16]. Tilings in which all rectangles are dyadic, but may have different areas, have been used as a basis for subdivision algorithms to solve problems such as approximating singular algebraic curves [2] and classifying data using decision trees [26]. In both of these examples, the unit square is repeatedly subdivided into smaller and smaller dyadic rectangles until the desired approximation or classification is achieved, with more subdivisions in the areas of the most interest (e.g., near the algebraic curve or where data classified differently is close together).

1.2 Proof ideas

We identify a certain block structure on dyadic tilings that allows us to relate the spectral gap of the edge-flip Markov chain to that of another, simpler Markov chain. In the simpler Markov chain, which we refer to as the block dynamics, for each transition a large region of the tiling is selected and retiled uniformly at random, if possible. At the smallest scale, $n = 4$, these correspond to exactly the moves of the (lazy) edge-flip Markov chain. The structure of these block moves allows us to set up a recursion that relates the spectral gap of the edge-flip Markov chain for tilings of size n with that of sizes smaller than n and that of the block dynamics. This produces an inverse polynomial lower bound on the spectral gap of the edge-flip Markov chain.

Specifically, we adapt a bisection approach inspired by spin system analysis [22, 6]. We bound the spectral gap γ_k of the Markov chain $\mathcal{M}_k$ for dyadic tilings of size $n = 2^k$ by the product of the spectral gap γ_{block} of the block dynamics Markov chain and the spectral gap γ_{k-1} of $\mathcal{M}_{k-1}$, and then use recursion to obtain $\gamma_k \geq (\gamma_{block})^k = (\gamma_{block})^{\log n}$. As γ_{block} is constant, this implies a polynomial relaxation time and thus a polynomial mixing time.

To establish the explicit upper bound in Theorem 1, we use a coupling argument to bound γ_{block}; see, e.g., Chapter 13 of [19]. The distance metric we use is a carefully weighted average of two different notions of distance between tilings. We do a case analysis and show this distance metric contracts by a factor of at least $1 - 1/17$ in each step, implying the spectral gap γ_{block} is at least $1/17$.

We use a distinguishing statistic to show the mixing time and relaxation time of the edge-flip Markov chain for dyadic tilings are at least $\Omega(n^{1.38})$; again, see Chapter 13 of [19]. That is, we define a specific function f on the state space of all dyadic tilings of size $n = 2^k$. By considering the variance and Dirichlet form of f, and using combinatorial properties of dyadic tilings, we can give an upper bound on the spectral gap and thus a lower bound on the relaxation and mixing times.

2 Background

Here we present some necessary information on dyadic tilings, including their asymptotic behavior, and on Markov chains, including mixing time and local variance.

2.1 Dyadic Tilings

A *dyadic interval* is an interval that can be written in the form $[a2^{-s}, (a+1)2^{-s}]$ for non-negative integers a and s with $0 \leq a < 2^s$. A *dyadic rectangle* is the product of two dyadic intervals. A *dyadic tiling of size* $n = 2^k$ is a tiling of the unit square by n dyadic rectangles of equal area $1/n = 2^{-k}$ that do not overlap except on their boundaries; see Figure 1. Let Ω_k be the set of all dyadic tilings of size $n = 2^k$. We say a dyadic tiling has a *vertical bisector* if the line $x = 1/2$ does not intersect the interior of any dyadic rectangle in the tiling. We say it has a *horizontal bisector* if the same is true of the line $y = 1/2$. It is easy to prove that every dyadic tiling of size $n > 1$ has a horizontal bisector or a vertical bisector.

The asymptotics of dyadic tilings were first explored by Lagarias, Spencer, and Vinson [17], and we present a summary of their results. Let $A_k = |\Omega_k|$ denote the number of dyadic tilings of size $n = 2^k$. The unit square is the unique dyadic tiling consisting of one dyadic rectangle, so $A_0 = 1$. There are two dyadic tilings of size 2, since the unit square may be divided by either a horizontal or vertical bisector, so $A_1 = 2$. One can also observe that $A_2 = 7$, $A_3 = 82$, $A_4 = 11047\ldots$. (The sequence appears in the Online Encyclopedia of Integer Sequences (OEIS) as A062764. [15]) In fact, the values A_k can be shown to satisfy the recurrence $A_k = 2A_{k-1}^2 - A_{k-2}^4$; we include a proof of this fact as presented in [16], because we will use these ideas later.

▶ **Proposition 3** ([17]). *For $k \geq 2$, the number of dyadic tilings of size 2^k is $A_k = 2A_{k-1}^2 - A_{k-2}^4$.*

Proof. A dyadic tiling of size 2^k has a horizontal bisector, a vertical bisector, or both. If it has a vertical bisector, the number of ways to tile the left half of the unit square is A_{k-1}; by mapping $x \to 2x$, we can see that the left half of a dyadic tiling of size 2^k is equivalent to a dyadic tiling of the unit square of size 2^{k-1} because dyadic rectangles scaled by factors of two remain dyadic. Similarly, mapping $x \to 2x - 1$, the right half of a dyadic tiling of size 2^k is equivalent to a dyadic tiling of size 2^{k-1}. We conclude the number of dyadic tilings of size 2^k with a vertical bisector is A_{k-1}^2. Similarly, by appealing to the maps $y \to 2y$ and $y \to 2y - 1$, the number of dyadic tilings of size 2^k with a vertical bisector is A_{k-1}^2. The number of dyadic tilings of size 2^k with both a horizontal and a vertical bisector is A_{k-2}^4, as each quadrant of any such tiling is equivalent to a dyadic tiling of size 2^{k-2}. This follows from appealing to the map $(x, y) \to (2x, 2y)$ for the lower left quadrant, and appropriate translations of this for the other three quadrants. Altogether, we see $A_k = A_{k-1}^2 + A_{k-1}^2 - A_{k-2}^4 = 2A_{k-1}^2 - A_{k-2}^4$. ◀

It is believed this recurrence does not have a closed form solution. As proved in [17], $A_k \sim \phi^{-1}\omega^{2^k} = \phi^{-1}\omega^n$, where $\phi = (1 + \sqrt{5})/2$ is the golden ratio and $\omega = 1.84454757...$; an exact value for ω is not known.

We now define a recurrence for another useful statistic. We say that a dyadic tiling has a *left half-bisector* if the straight line segment from $(0, 1/2)$ to $(1/2, 1/2)$ doesn't intersect the interior of any dyadic rectangles. Figure 1(a) does not have a left half-bisector, while Figure 1(b) does. We are interested in the number of ways to tile the left half of a vertically-bisected dyadic tiling of size 2^k such that it has a left half-bisector. Appealing to the dilation maps defined in the proof of Proposition 3, this number is A_{k-2}^2. Among all possible ways to tile the left half of a vertically-bisected tiling $\sigma \in \Omega_k$, we define f_k to be the fraction with a left half-bisector. We see

$$f_k = \frac{A_{k-2}^2}{A_{k-1}}.$$

We can similarly define *right half-bisectors*, *top half-bisectors*, and *bottom half-bisectors* by considering the straight line segments between $(1/2, 1/2)$ and, respectively, $(1, 1/2)$, $(1/2, 1)$, and $(1/2, 0)$. Then f_k is also the fraction of tilings of the right half of vertically-bisected tiling σ with a right half-bisector, or the fraction of tilings of the top or bottom halves of a horizontally-bisected tiling σ with a top or bottom half-bisector, respectively. Note $f_2 = 0.5$, $f_3 = 4/7 \sim 0.571$, and $f_4 = 49/82 \sim 0.598$. We now examine the asymptotic behavior of f_k; the following lemmas are proved in Section B.

▶ **Lemma 4.** *For all $k \geq 3$, $f_k = \frac{1}{2 - f_{k-1}^2}$.*

▶ **Lemma 5.** *The sequence* $\{f_k\}_{k=2}^{\infty}$ *is strictly increasing and bounded above by* $(\sqrt{5}-1)/2$*. Furthermore,* $\lim_{k\to\infty} f_k = (\sqrt{5}-1)/2$*.*

2.2 Markov Chains

We will consider only discrete time Markov chains in this paper, though identical results hold for the analogous continuous time Markov chains. Any finite ergodic Markov chain converges to a unique stationary distribution π. The time a Markov chain with transition matrix P takes to converge to its stationary distribution is measured by the *total variation distance*, which captures how far the distribution after t steps is from the stationary distribution given a worst case starting configuration:

$$\|P^t - \pi\|_{\mathrm{TV}} = \max_{x\in\Omega} \frac{1}{2} \sum_{y\in\Omega} |P^t(x,y) - \pi(y)|.$$

The mixing time of a Markov chain $\mathcal{M}$ is defined to be

$$t_{\mathrm{mix}}(\varepsilon) = \min\{t : \|P^{t'} - \pi\|_{\mathrm{TV}} \le \varepsilon \ \ \forall\, t' \ge t\}.$$

For convenience, as is standard we define $t_{\mathrm{mix}} = t_{\mathrm{mix}}(1/4)$.

We will bound the mixing time of the edge-flip Markov chain for dyadic tilings by studying its relaxation time and spectral gap. The *spectral gap* γ of a Markov chain $\mathcal{M}$ with transition matrix P is $1-\lambda_2$, where λ_2 is the second largest eigenvalue of P. A *lazy* Markov chain is one where $P(x,x) \ge 1/2$ for all $x \in \Omega$; for a lazy Markov chain $\mathcal{M}$, the relaxation time, denoted by t_{rel}, is then the inverse of this spectral gap. We will see in the next section that the edge-flip Markov chain for dyadic tilings is lazy. The following well-known proposition relates the relaxation time and mixing time for Markov chains; for a proof, see, e.g., [19, Theorem 12.3 and Theorem 12.4].

▶ **Proposition 6.** *Let* $\mathcal{M}$ *be an ergodic Markov chain on state space* Ω *with reversible transition matrix* P *and stationary distribution* π*. Let* $\pi_{min} = \min_{x\in\Omega} \pi(x)$*. Then:*

$$(t_{\mathrm{rel}} - 1)\log\left(\frac{1}{2\varepsilon}\right) \le t_{\mathrm{mix}}(\varepsilon) \le \log\left(\frac{1}{\varepsilon\pi_{min}}\right) t_{\mathrm{rel}}.$$

We will bound the spectral gap, and thus the relaxation and mixing times, of the edge-flip Markov chain for dyadic tilings by considering functions on the chain's state space. For $f : \Omega \to \mathbb{R}$, the *variance* of f with respect to a distribution π on Ω can be expressed as:

$$\mathrm{var}_\pi(f) = \sum_{x\in\Omega} \pi(x)\left(f(x) - \mathbb{E}_\pi[f(x)]\right)^2 = \frac{1}{2}\sum_{x,y\in\Omega} \pi(x)\pi(y)(f(x)-f(y))^2.$$

We will only be considering the variance with respect to the uniform distribution on Ω, so the subscript π will be omitted. For a given reversible transition matrix P on state space Ω with stationary distribution π, the *Dirichlet form*, also known as the *local variance*, associated to the pair (P,π) is, for any function $f : \Omega \to \mathbb{R}$,

$$\mathcal{E}(f) = \frac{1}{2}\sum_{x,y\in\Omega} [f(x)-f(y)]^2 \pi(x) P(x,y).$$

As we see in the following well-known proposition, the Dirichlet form and variance of a function f can be used to bound the spectral gap of a transition matrix, and therefore the relaxation time and mixing time of a Markov chain; see, e.g., [19, Lemma 13.12].

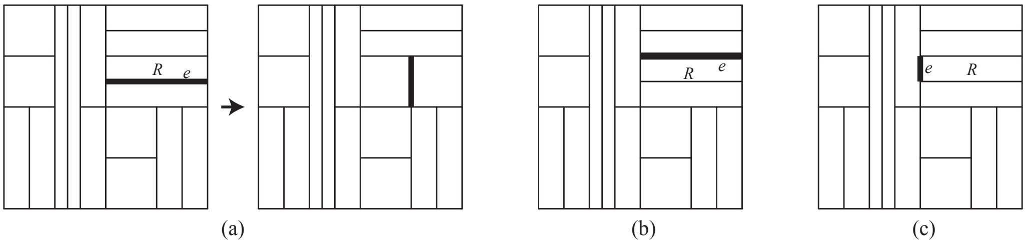


Figure 2 A random rectangle R and one of its edges e are selected in each iteration of $\mathcal{M}_k$. (a) Random choices of R and e as shown yield a valid edge flip. (b) Random choices of R and e as shown do not yield a valid edge flip as flipping edge e results in a tiling that is not dyadic. (c) Random choices of R and e as shown do not yield a valid edge flip as flipping edge e does not produce a tiling of the unit square by rectangles.

▶ **Proposition 7.** *Given a Markov chain with reversible transition matrix P and stationary distribution π, the spectral gap $\gamma = 1 - \lambda_2$ of P satisfies*

$$\gamma = \min_{\substack{f:\Omega\to\mathbb{R} \\ \mathrm{var}_\pi(f)\neq 0}} \frac{\mathcal{E}(f)}{\mathrm{var}_\pi(f)}.$$

3 The Edge-Flip Markov Chain $\mathcal{M}_k$

Let $n = 2^k$. For $k \geq 1$, the edge-flip Markov chain $\mathcal{M}_k$ on the state space Ω_k of all dyadic tilings of size 2^k is given by the following rules.

Beginning at any $\sigma_0 \in \Omega_k$, repeat:

- Choose a rectangle R of σ_i uniformly at random.
- Choose *left*, *right*, *top*, or *bottom* uniformly at random; let e be the corresponding side of R.
- If e bisects a rectangle of area 2^{-k+1}, remove e and replace it with its perpendicular bisector to obtain σ_{i+1} if the result is a valid dyadic tiling; else, set $\sigma_{i+1} = \sigma_i$.

An example of an edge-flip move of $\mathcal{M}_k$ is shown in Figure 2(a); two selections of R and e that do not yield valid moves are shown in (b) and (c). Let $P_{k,edge}$ denote the transition matrix of this edge-flip Markov chain and γ_k its spectral gap. For every valid edge flip, there are two choices of (R, e) that produce that move. This implies every move between two tilings differing by an edge flip occurs with probability $1/(2n) = 2^{-k-1}$, so all off-diagonal entries of $P_{k,edge}$ are 2^{-k-1} or 0.

The Markov chain $\mathcal{M}_k$, in a slightly different form, was introduced by Janson, Randall and Spencer [16]. Note $\mathcal{M}_k$ is lazy, as for any rectangle R of a dyadic tiling at most one of its left and right edges can be flipped to produce another valid dyadic tiling. This is because if R's projection onto the x-axis is dyadic interval $[a2^{-s}, (a+1)2^{-s}]$ for $a, s \in \mathbb{Z}_{\geq 0}$, then flipping its left edge yields a rectangle with x-projection $[(a-1)2^{-s}, (a+1)2^{-s}]$ and flipping its right edge yields a rectangle with x-projection $[a2^{-s}, (a+2)2^{-s}]$. If a is even, the first of these intervals is not dyadic, while if a is odd, the second is not, so at most one of these edge flips produces a valid dyadic tiling. Similarly, at most one of R's top and bottom edges yields a valid edge flip. This implies in each iteration with probability at least 1/2 a pair (R, e) is selected that does not yield a valid edge flip move.

It was previously shown that this Markov chain is irreducible [16], so $\mathcal{M}_k$ is ergodic and thus has a unique stationary distribution. The uniform distribution satisfies the detailed balance equation, implying both that $\mathcal{M}_k$ is reversible and that its stationary distribution is uniform on Ω_k.

While we index this edge-flip Markov chain for dyadic tilings of size $n = 2^k$ by k instead of by n, note we wish to show the mixing time of $\mathcal{M}_k$ is polynomial in n, not polynomial in k.

3.1 The Block Dynamics Markov Chain $\mathcal{M}_k^{block}$

To analyze the mixing time of Markov chain $\mathcal{M}_k$, we will appeal to a similar Markov chain that uses larger block moves instead of single edge flips. We use in a crucial way the bijection between tilings in Ω_{k-1} and the left or right (resp. top or bottom) half of a tiling in Ω_k that has a vertical (resp. horizontal) bisector, as discussed in the proof of Proposition 3. For $k \geq 2$, the block dynamics Markov chain $\mathcal{M}_k^{block}$ on the state space Ω_k of all dyadic tilings of size 2^k is given by the following rules.

Beginning at any dyadic tiling σ_0, repeat:

- Uniformly at random choose a tiling $\rho \in \Omega_{k-1}$.
- Uniformly at random choose *Left*, *Right*, *Top*, or *Bottom*.
- To obtain σ_{i+1}:
  - If *Left* was chosen and σ has a vertical bisector, retile σ's left half with ρ, under the mapping $x \to x/2$.
  - If *Right* was chosen and σ has a vertical bisector, retile σ's right half with ρ, under the mapping $x \to (x+1)/2$.
  - If *Bottom* was chosen and σ has a horizontal bisector, retile σ's bottom half with ρ, under the mapping $y \to y/2$.
  - If *Top* was chosen and σ has a horizontal bisector, retile σ's top half with ρ, under the mapping $y \to (y+1)/2$.
- Else, set $\sigma_{i+1} = \sigma_i$.

Let $P_{k,block}$ be the transition matrix of this Markov chain and let $\gamma_{k,block}$ be its spectral gap. Any valid nonstationary transition of $\mathcal{M}_k^{block}$ occurs with probability $1/(4|\Omega_{k-1}|)$. This Markov chain is not lazy, but it is aperiodic, irreducible, and reversible. This implies it is ergodic and thus has a unique stationary distribution, which by detailed balance is uniform on Ω_k.

4 A Polynomial upper bound on the mixing time of $\mathcal{M}_k$

Recall we wish to show the mixing time of $\mathcal{M}_k$ is polynomial in $n = 2^k$, not polynomial in k. We show the spectral gap γ_k of $\mathcal{M}_k$ and the spectral gap γ_{k-1} of $\mathcal{M}_{k-1}$ differ by a multiplicative constant (specifically, 1/17) by appealing to the Dirichlet forms of both of these Markov chains as well as the block dynamics Markov chain $\mathcal{M}_k^{block}$. We can then use recursion to show γ_k is bounded below by $(1/17)^k$, which, because $k = \log n$, gives a polynomial upper bound on the relaxation time and thus on the mixing time of $\mathcal{M}_k$.

For any function $f : \Omega_k \to \mathbb{R}$, we will denote the Dirichlet form of f with respect to transition matrix $P_{k,edge}$ and the uniform stationary distribution as $\mathcal{E}_{k,edge}(f)$. The Dirichlet form of f with respect to transition matrix $P_{k,block}$ and the uniform stationary distribution will be $\mathcal{E}_{k,block}(f)$. We will let the variance of function f on Ω_k with respect to the uniform stationary distribution be $\mathrm{var}_k(f)$. Here the k indicates which state space Ω_k we are considering, rather than which distribution on Ω_k the variance is taken with respect to; all variances we consider will be with respect to the uniform distribution.

Because we consider two different Markov chains on the same state space Ω_k, there are two different notions of adjacencies on this state space, each corresponding to the moves of one of these Markov chains. For $x, y \in \Omega_k$, we say $x \sim_e y$ if x and y differ by a single edge flip move of $\mathcal{M}_k$ and $x \sim_b y$ if x and y differ by a single move of the block dynamics chain $\mathcal{M}_k^{block}$. More specifically, if x and y differ by a retiling of their left half (implying x and y both have a vertical bisector and are the same on their right half), we say $x \sim_L y$; then $x \sim_R y$, $x \sim_T y$, and $x \sim_B y$ are defined similarly for the right, top, and bottom halves.

▶ **Theorem 8.** *For any $k \geq 2$, the spectral gap γ_k of the edge-flip Markov chain $\mathcal{M}_k$ satisfies*

$$\gamma_k \geq \gamma_{k,block} \cdot \gamma_{k-1}$$

Proof. We begin by relating the Dirichlet forms for block dynamics and for the edge-flip dynamics, which will allow comparison of their spectral gaps. Recall that for any function $f : \Omega_k \to \mathbb{R}$,

$$\mathcal{E}_{k,block}(f) = \frac{1}{2} \sum_{x \sim_b y \in \Omega_k} \pi(x) P_{k,block}(x, y) \left(f(x) - f(y)\right)^2 .$$

This sum can be split into four terms, corresponding to the type of block move (left, right, top, or bottom) transforming x into y. If x and y differ only in their top-left quadrants, then x could transition to y via either a left block move or a top block move; each of these moves occurs with probability $\frac{1}{4|\Omega_{k-1}|}$, and the total probability of $P_{k,block}(x, y) = \frac{1}{2|\Omega_{k-1}|}$ will be split correspondingly between the terms for left block moves and top block moves.

We now analyze the first of these terms, containing all x, y differing by a retiling of their left halves. For $x_L, x_R \in \Omega_{k-1}$, by $x_L x_R$ below we mean the tiling in Ω_k with a vertical bisector whose left half is x_L under the map $x \to x/2$ and whose right half is x_R under the map $x \to (x + 1)/2$.

$$\begin{aligned}
\mathcal{E}^L_{k,block}(f) &= \frac{1}{2} \sum_{x \sim_L y} \frac{1}{|\Omega_k|} \frac{1}{4|\Omega_{k-1}|} (f(x) - f(y))^2 \\
&= \frac{1}{8} \sum_{x_R \in \Omega_{k-1}} \sum_{x_L, y_L \in \Omega_{k-1}} \frac{1}{|\Omega_k|} \frac{1}{|\Omega_{k-1}|} (f(x_L x_R) - f(y_L x_R))^2 \\
&= \frac{1}{4} \sum_{x_R \in \Omega_{k-1}} \frac{|\Omega_{k-1}|}{|\Omega_k|} \left(\frac{1}{2} \sum_{x_L, y_L \in \Omega_{k-1}} \frac{1}{|\Omega_{k-1}|^2} (f(x_L x_R) - f(y_L x_R))^2 \right) .
\end{aligned}$$

We note that the second sum above is over all pairs of tilings in Ω_{k-1}. While the Dirichlet form of a function sums over all pairs of states that differ by a transition of a Markov chain, the variance of a function sums over all pairs of states, regardless of the local structure imposed on the state space by the Markov chain. In fact, we have written the second sum above suggestively, and note that it is in fact a variance of a function over the state space Ω_{k-1}. For each $x_R \in \Omega_{k-1}$, the function $f|_{x_R} : \Omega_{k-1} \to \mathbb{R}$ given by $f|_{x_R}(z) = f(z x_R)$ has variance $\text{var}_{k-1}(f|_{x_R})$ (with respect to the uniform distribution) that is exactly equal to the term in parentheses above. Because the variance of a function is the same regardless of which transitions on the state space we are considering, it is through this variance we can relate $\mathcal{E}_{k,block}$, which we have calculated above, to a Dirichlet form for edge-flip moves. That is, by Proposition 7, we can bound this variance with the Dirichlet form of $f|_{x_R}$ associated to $P_{k-1,edge}$ and the spectral gap γ_{k-1} of $\mathcal{M}_{k-1}$. Thus,

$$\mathcal{E}^L_{k,block}(f) = \frac{1}{4} \sum_{x_R \in \Omega_{k-1}} \frac{|\Omega_{k-1}|}{|\Omega_k|} \text{var}_{k-1}(f|_{x_R}) \leq \frac{1}{4} \sum_{x_R \in \Omega_{k-1}} \frac{|\Omega_{k-1}|}{|\Omega_k|} \frac{\mathcal{E}_{k-1,edge}(f|_{x_R})}{\gamma_{k-1}} .$$

We now see that the Dirichlet form for the edge-flip Markov chain on Ω_{k-1} is

$$\begin{aligned}\mathcal{E}_{k-1,edge}(f|_{x_R}) &= \frac{1}{2}\sum_{\substack{x_L,y_L\in\Omega_{k-1}\\ x_L\sim_e y_L}} \pi(x_L)P(x_L,y_L)\left(f(x_Lx_R)-f(y_Lx_R)\right)^2 \\ &= \sum_{\substack{x_L,y_L\in\Omega_{k-1}\\ x_L\sim_e y_L}} \frac{1}{|\Omega_{k-1}|}\frac{1}{2n}\left(f(x_Lx_R)-f(y_Lx_R)\right)^2 .\end{aligned}$$

Using this expression, we see that

$$\begin{aligned}\mathcal{E}^L_{k,block}(f) &\le \frac{1}{4\gamma_{k-1}}\sum_{x_R\in\Omega_{k-1}}\frac{|\Omega_{k-1}|}{|\Omega_k|}\left(\sum_{\substack{x_L,y_L\in\Omega_{k-1}\\ x_L\sim_e y_L}}\frac{1}{|\Omega_{k-1}|}\frac{1}{2n}\left(f(x_Lx_R)-f(y_Lx_R)\right)^2\right) \\ &= \frac{1}{4\gamma_{k-1}}\sum_{\substack{x,y\in\Omega_k\\ x\sim_e y\\ x\sim_L y}}\frac{1}{|\Omega_k|}\frac{1}{2n}\left(f(x)-f(y)\right)^2 .\end{aligned}$$

We now compare this to the Dirichlet form for the edge flip Markov chain on Ω_k, which we recall is

$$\mathcal{E}_{k,edge}(f) = \frac{1}{2}\sum_{\substack{x,y\in\Omega_k\\ x\sim_e y}}\frac{1}{|\Omega_k|}\frac{1}{2n}\left(f(x)-f(y)\right)^2 .$$

We note for every $x,y\in\Omega_k$ such that $x\sim_e y$, at least one of and at most two of $x\sim_L y$, $x\sim_R y$, $x\sim_T y$, and $x\sim_B y$ hold. Thus each summand of $\mathcal{E}_{k,edge}(f)$ appears at most twice as a summand of

$$\mathcal{E}_{k,block}(f) = \mathcal{E}^L_{k,block}(f)+\mathcal{E}^R_{k,block}(f)+\mathcal{E}^T_{k,block}(f)+\mathcal{E}^B_{k,block}(f).$$

It follows that

$$\mathcal{E}_{k,block}(f) \le \frac{1}{4\gamma_{k-1}}\cdot 2\cdot\left(2\mathcal{E}_{k,edge}(f)\right) = \frac{\mathcal{E}_{k,edge}(f)}{\gamma_{k-1}}.$$

Note this implies that for any f,

$$\mathrm{var}_k(f) \le \frac{\mathcal{E}_{k,block}(f)}{\gamma_{k,block}} \le \frac{\mathcal{E}_{k,edge}(f)}{\gamma_{k,block}\cdot\gamma_{k-1}}.$$

Let f be chosen to be the function achieving equality in $\mathrm{var}_k(f)\le\frac{\mathcal{E}_{k,edge}(f)}{\gamma_k}$. We conclude

$$\gamma_k = \frac{\mathcal{E}_{k,edge}(f)}{\mathrm{var}_k(f)} \ge \gamma_{k,block}\cdot\gamma_{k-1}. \qquad ◀$$

In Section A we prove that $\gamma_{k,block}$ is at least $1/17$ for sufficiently large k. This can be used to bound the spectral gap, the relaxation time, and finally the mixing time of $\mathcal{M}_k$.

▶ **Theorem 9.** *There exists a positive integer k_0 such that for all $k\ge k_0$, $\gamma_{k,block}\ge 1/17$.*

Proof. See Section A. We introduce a distance metric on dyadic tilings, and then give a coupling where the distance between two tilings decreases in expectation after one iteration by a multiplicative factor of $1-\frac{1}{17}$ for all k sufficiently large. By a result of Chen [7] (see also [19, Theorem 13.1]), this implies the theorem. ◀

We are now ready to prove our first main theorem, Theorem 1, which states that the relaxation time of $\mathcal{M}_k$ for $n = 2^k$ is $O(n^{\log 17})$ and its mixing time is $O(n^{1+\log 17})$

Proof of Theorem 1. By Theorems 8 and 9, the spectral gap of $\mathcal{M}_k$ satisfies

$$\gamma_k \geq \frac{1}{17}\gamma_{k-1} \geq 17^{-(k-k_0)}\gamma_{k_0},$$

where k_0 is the value from Theorem 9. Since γ_{k_0} is a constant that does not depend on n,

$$\gamma_k = \Omega\left(17^{-k}\right) = \Omega\left(n^{-\log 17}\right) = \Omega\left(n^{-4.09}\right).$$

Because $\mathcal{M}_k$ is a lazy Markov chain, its relaxation time satisfies

$$t_{\text{rel}} = O\left(n^{\log 17}\right).$$

To use this to bound the mixing time of $\mathcal{M}_k$, we appeal to Proposition 6, though we first must calculate π_{min}. For π the uniform distribution, $\min_{x \in \Omega_k} \pi(x) = 1/|\Omega_k|$. By Proposition 3, $|\Omega_k| < 2|\Omega_{k-1}|^2$, so a loose bound is $1/\pi_{min} = |\Omega_k| < 2^{2^k} = 2^n$. This implies

$$t_{\text{mix}} = O\left(n^{1+\log 17}\right). \qquad ◀$$

5 Lower bound on the mixing time of $\mathcal{M}_n$

In this section we give the proof of Theorem 2. For this, we define the following subsets of Ω_k:

$$\begin{aligned} \Omega_k^+ &= \{x \in \Omega_k : \ x \text{ has both a horizontal and a vertical bisector}\}, \\ \Omega_k^| &= \{x \in \Omega_k : \ x \text{ has a vertical bisector}\}, \text{ and} \\ \Omega_k^- &= \{x \in \Omega_k : \ x \text{ has a horizontal bisector}\}. \end{aligned}$$

By definition, we have $\Omega_k^+ = \Omega_k^| \cap \Omega_k^-$. We start with the following simple lemma.

▶ **Lemma 10.** *For all $k \geq 2$, we have*

$$\frac{|\Omega_k|}{|\Omega_k^+|} = \frac{2}{f_k^2} - 1 \geq 2\phi + 1,$$

where $\phi = \frac{\sqrt{5}+1}{2}$ is the golden ratio. Furthermore, $\lim_{k\to\infty} \frac{|\Omega_k|}{|\Omega_k^+|} = 2\phi + 1$.

Proof. Using that $|\Omega_k^+| = |\Omega_{k-2}|^4$, and Proposition 3, we have

$$\frac{|\Omega_k|}{|\Omega_k^+|} = \frac{2|\Omega_{k-1}|^2 - |\Omega_{k-2}|^4}{|\Omega_{k-2}|^4} = \frac{2}{f_k^2} - 1.$$

By Lemma 5, $f_k \leq \frac{\sqrt{5}-1}{2} = \frac{1}{\phi} = \lim_{k\to\infty} f_k$. This, along with the identity $\phi^2 = 1+\phi$, implies the lemma. ◀

We will also require the following technical estimate.

▶ **Lemma 11.** *For any $k \geq 2$, we have*

$$\frac{1}{|\Omega_k|}\prod_{i=0}^{k-2} |\Omega_i|^2 \leq \phi^{-2k+2}.$$

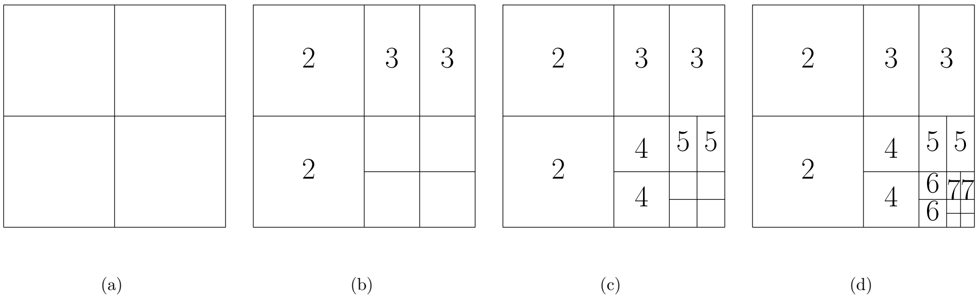


Figure 3 The construction of a tiling to count $\prod_{i=0}^{k-2} |\Omega_i|^2$. A rectangle with number a indicates that we tile it with a tiling from Ω_{k-a}.

Proof. We will show how to estimate $\prod_{i=0}^{k-2} |\Omega_i|^2$ via the construction of a tiling in Ω_k. We start with a tiling with both a horizontal and a vertical bisector, as in Figure 3(a). Then we inductively do the following. Both quadrants of the left half are tiled independently with a uniformly random tiling from Ω_{k-2}. In the top-right quadrant, we add a vertical bisector and complete the two halves of this quadrant with independent, uniformly random tilings from Ω_{k-3}. Finally, in the bottom-right quadrant, we create a horizontal and a vertical bisector, reaching the tiling in Figure 3(b). Then we take this bottom-right quadrant, and iterate the procedure above; see Figure 3(c,d) for the configurations after one and two more iterations.

This iteration continues until creating a bisector will result in rectangles of area less than 2^{-k}. In the case where an attempt is made to divide a rectangle of area 2^{-k+1} into four rectangles of equal area by adding both a horizontal and vertical bisector, we instead add just a horizontal bisector, resulting in two rectangles each of area 2^{-k}.

Let $\Upsilon_k \subset \Omega_k$ be the set of tilings obtained in this way. Note that the number of tilings in Υ_k is exactly $\prod_{i=0}^{k-2} |\Omega_i|^2$. Since $\Upsilon_k \subset \Omega_k^+$, we have that $\frac{|\Upsilon_k|}{|\Omega_k|} \leq \frac{|\Omega_k^+|}{|\Omega_k|}$, where the first expression is exactly the value we wish to bound. Using the construction above until Figure 3(b), we obtain that

$$\frac{|\Upsilon_k|}{|\Omega_k|} \leq \frac{|\Omega_k^+|}{|\Omega_k|} \frac{|\Omega_{k-2}^{|}|}{|\Omega_{k-2}|},$$

where the second factor stands for the fact that the top-right quadrant must contain a vertical bisector. Iterating this in the bottom-right quadrant, we obtain

$$\frac{|\Upsilon_k|}{|\Omega_k|} \leq \frac{|\Omega_k^+|}{|\Omega_k|} \frac{|\Omega_{k-2}^{|}|}{|\Omega_{k-2}|} \frac{|\Omega_{k-2}^+|}{|\Omega_{k-2}|} \frac{|\Omega_{k-4}^{|}|}{|\Omega_{k-4}|} \cdots \tag{1}$$

Proposition 3 gives that

$$\frac{|\Omega_k^{|}|}{|\Omega_k|} = \frac{|\Omega_k| + |\Omega_{k-2}|^4}{2|\Omega_k|} = \frac{1}{2}\left(1 + \frac{|\Omega_k^+|}{|\Omega_k|}\right) \leq \frac{1}{2}\left(1 + \frac{1}{2\phi+1}\right) = \frac{\phi^2}{2\phi+1},$$

where the inequality follows from Lemma 10. For even k, because $|\Omega_0^{|}| = 0$ the last term we can obtain in (1) is $\frac{|\Omega_2^+|}{|\Omega_2|}$, so we can write

$$\begin{aligned}\frac{|\Upsilon_k|}{|\Omega_k|} &\leq \left(\prod_{i=0}^{k/2-2} \frac{|\Omega_{k-2i}^+|}{|\Omega_{k-2i}|} \cdot \frac{|\Omega_{k-2i-2}^{|}|}{|\Omega_{k-2i-2}|}\right) \frac{|\Omega_2^+|}{|\Omega_2|} \\ &\leq \frac{1}{2\phi+1}\left(\frac{1}{2\phi+1} \cdot \frac{\phi^2}{2\phi+1}\right)^{\frac{k}{2}-1} = \frac{\phi^{-2k+4}}{2\phi+1} \leq \phi^{-2k+2},\end{aligned}$$

where the last expressions come from, respectively, identities for ϕ and the easily-checked inequality $2\phi + 1 > \phi^2$. When k is odd, the last term in (1) is $\frac{|\Omega^{|}_1|}{|\Omega_1|}$ because $|\Omega^{+}_1| = 0$, so we can write

$$\frac{|\Upsilon_k|}{|\Omega_k|} \leq \left(\prod_{i=0}^{(k-3)/2} \frac{|\Omega^{+}_{k-2i}|}{|\Omega_{k-2i}|} \cdot \frac{|\Omega^{|}_{k-2i-2}|}{|\Omega_{k-2i-2}|} \right) \leq \left(\frac{1}{2\phi+1} \cdot \frac{\phi^2}{2\phi+1} \right)^{\frac{k-1}{2}} \leq \phi^{-2k+2},$$

where again the last expression is the result of applying identities for ϕ and simplifying. ◀

We are now ready to prove our second main theorem, giving a lower bound on the mixing and relaxation times of $\mathcal{M}_k$ of $\Omega(n^{2\log\phi})$.

Proof of Theorem 2. We will derive a upper bound on the spectral gap γ_k. To do this, we consider the test function $f : \Omega_k \to \{0, 1\}$ such that

$$f(x) \text{ is } 1 \text{ if } x \in \Omega^{|}_k, \text{ and } 0 \text{ otherwise.} \tag{2}$$

We will apply this function to the characterization of the spectral gap in Proposition 7.

We start by showing that the variance of f is bounded away from 0 as $k \to \infty$. Recall that var_k denotes variance with respect to the uniform measure on Ω_k.

▶ **Claim 12.** *With $f : \Omega_k \to \{0, 1\}$ as in (2), we have that*

$$\lim_{k\to\infty} \mathrm{var}_k(f) = \sqrt{5} - 2.$$

Proof of Claim. We start by writing

$$\mathrm{var}_k(f) = \sum_{x\in\Omega^{|}_k} \sum_{y\in\Omega_k\setminus\Omega^{|}_k} \frac{1}{|\Omega_k|^2} = \frac{|\Omega^{|}_k| \cdot |\Omega_k \setminus \Omega^{|}_k|}{|\Omega_k|^2}. \tag{3}$$

Since $|\Omega^{|}_k| = |\Omega_{k-1}|^2$, using Proposition 3 we obtain

$$|\Omega^{|}_k| = \frac{|\Omega_k| + |\Omega_{k-2}|^4}{2} = \frac{|\Omega_k| + |\Omega^{+}_k|}{2}, \tag{4}$$

and

$$|\Omega_k \setminus \Omega^{|}_k| = |\Omega_k| - |\Omega^{|}_k| = \frac{|\Omega_k| - |\Omega^{+}_k|}{2}. \tag{5}$$

Plugging (4) and (5) into (3), we get

$$\mathrm{var}_k(f) = \frac{1}{4}\left(1 + \frac{|\Omega^{+}_k|}{|\Omega_k|}\right)\left(1 - \frac{|\Omega^{+}_k|}{|\Omega_k|}\right) = \frac{1}{4}\left(1 - \left(\frac{|\Omega^{+}_k|}{|\Omega_k|}\right)^2\right).$$

Then Lemma 10 yields

$$\lim_{k\to\infty} \mathrm{var}_k(f) = \frac{1}{4}\left(1 - \frac{1}{(2\phi+1)^2}\right).$$

Plugging in the value of ϕ completes the proof of the claim. ◀

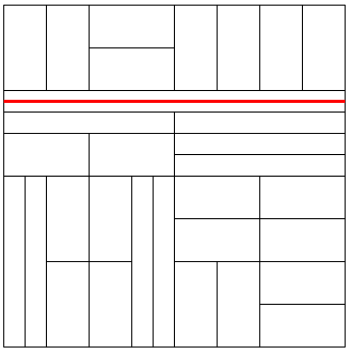


Figure 4 A tiling in $\partial\Omega_k^{|}$, with the red edge being the flip that brings the tiling into $\Omega_k^{|}$.

Now it remains to obtain an upper bound for $\mathcal{E}(f)$. Let $\partial\Omega_k^{|}$ be the set of tilings in $\Omega_k \setminus \Omega_k^{|}$ which can be obtained from a tiling in $\Omega_k^{|}$ via one edge flip. Recall for two tilings $x, y \in \Omega_k$, we write $x \sim_e y$ if x can be obtained from y by one edge flip. Hence,

$$\mathcal{E}(f) = \sum_{x \in \partial\Omega_k^{|}} \sum_{y \in \Omega_k^{|}:\, y \sim_e x} \frac{1}{|\Omega_k|} \frac{1}{2n}.$$

Note that each tiling in $\partial\Omega_k^{|}$ has a horizontal bisector and is not in Ω_k^{+}. This means that it has exactly one edge flip that can bring it into $\Omega_k^{|}$, which is the flip that creates a vertical bisector. Then, we have

$$\mathcal{E}(f) = \frac{|\partial\Omega_k^{|}|}{2n \cdot |\Omega_k|}.$$

Now we need to describe the set $\partial\Omega_k^{|}$. It is a set of tilings with no vertical bisector, but with one edge flip that creates a vertical bisector; see Figure 4.

Note that the edge whose flip creates a vertical bisector must be a horizontal edge of length 1 which flips to a vertical edge of length $2/n$. From now on we will refer to this edge as the *pivotal edge.*

In order to estimate the cardinality of $\partial\Omega_k^{|}$, we will describe a procedure to construct a tiling $x \in \partial\Omega_k^{|}$, observing the position of the pivotal edge. Note that x must have a horizontal bisector, which splits $[0,1]^2$ into its top and bottom halves. Assume that the pivotal edge is in the top half of x. This implies that the bottom half of x must itself contain a vertical bisector since the pivotal edge must be the only edge that forbids a vertical bisector to exist, see Figure 5(a). The two quadrants in the bottom half are simply any tilings of Ω_{k-2}. Note also that the top half of x must contain a horizontal bisector, otherwise $x \notin \partial\Omega_k^{|}$, see Figure 5(b). Then we iterate the above construction: among the two halves of the top half, one must contain the pivotal edge, say the bottom one, while the other contains a vertical bisector, each side of which being completed with a tiling from Ω_{k-3}, which gives the configuration in Figure 5(c). Continuing this for $k-2$ steps concludes the construction.

To estimate the cardinality of $\partial\Omega_k^{|}$, note that in each step of the construction we have two choices for where the pivotal edge is: either in the top half or the bottom half of the corresponding region. Therefore, the number of tilings in $\partial\Omega_k^{|}$ is

$$|\partial\Omega_k^{|}| = \prod_{i=2}^{k} \left(2|\Omega_{k-i}|^2\right) = 2^{k-1} \prod_{i=0}^{k-2} |\Omega_i|^2 = \frac{n}{2} \prod_{i=0}^{k-2} |\Omega_i|^2.$$

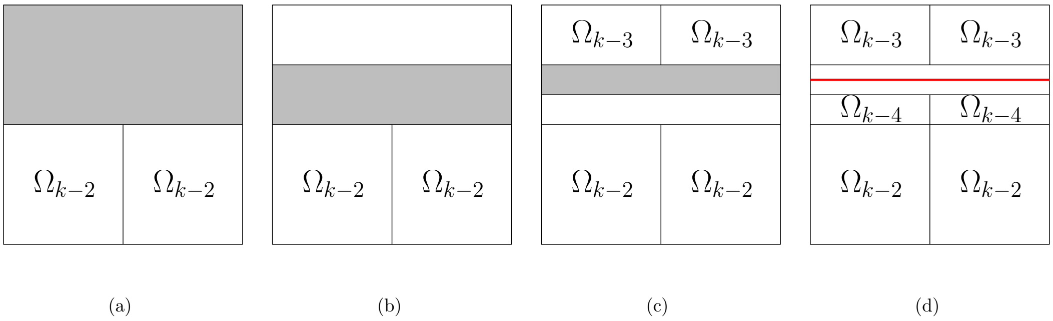


Figure 5 The construction of a tiling in $\partial\Omega_k^|$. The grey areas represent the part that contains the pivotal edge.

Hence,

$$\mathcal{E}(f) = \frac{1}{4|\Omega_k|} \prod_{i=0}^{k-2} |\Omega_i|^2 \leq \frac{1}{4}\phi^{-2k+2}$$

where the last step follows from Lemma 11. Therefore, there exists a constant $c > 0$ such that

$$\gamma_k \leq c\phi^{-2k}.$$

This implies that the relaxation time and mixing time satisfy

$$t_{\text{rel}}, t_{\text{mix}} \geq \frac{1}{c}\phi^{2k} = \frac{1}{c}\phi^{2\log n} = \frac{1}{c}n^{2\log\phi} = \Omega(n^{2\log\phi}).$$

This completes the proof of the theorem. ◀

Acknowledgements. This work started during the 2016 AIM workshop *Markov chain mixing times.* We thank the organizers for the invitation and the stimulating atmosphere.

References

1 Russ Bubley and Martin Dyer. Path coupling: A technique for proving rapid mixing in markov chains. In *FOCS'97: Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS)*, 1997.

2 Michael Burr, Sung Woo Choi, Ben Galehouse, and Chee K. Yap. Complete subdivision algorithms, II: Isotopic meshing of singular algebraic curves. *Journal of Symbolic Computation*, 47(2):131–152, 2012. doi:10.1016/j.jsc.2011.08.021.

3 Sarah Cannon, Sarah Miracle, and Dana Randall. Phase transitions in random dyadic tilings and rectangular dissections. In *Proceedings of the 26th Symposium on Discrete Algorithms (SODA)*, 2015.

4 Pietro Caputo, Fabio Martinelli, Alistair Sinclair, and Alexandre Stauffer. Random lattice triangulations: Structure and algorithms. *The Annals of Applied Probability*, 25(4):1650–1685, 2015.

5 Pietro Caputo, Fabio Martinelli, Alistair Sinclair, and Alexandre Stauffer. Dynamics of lattice triangulations on thin rectangles. *Electronic Journal of Probability*, 21(29), 2016.

6 Filippo Cesi. Quasi-factorization of the entropy and logarithmic Sobolev inequalities for Gibbs random fields. *Probability Theory and Related Fields*, 120(4):569–584, 2001. doi:10.1007/PL00008792.

7 Mu-Fa Chen. Trilogy of couplings and general formulas for lower bound of spectral gap. In *Probability towards 2000 (New York, 1995)*, volume 128 of *Lecture Notes in Statist.*, pages 123–136. Springer, New York, 1998. doi:10.1007/978-1-4612-2224-8_7.

8 Colin Cooper, Martin Dyer, and Catherine Greenhill. Sampling regular graphs and a peer-to-peer network. *Combinatorics, Probability and Computing*, 16(4):557–593, July 2007. doi:10.1017/S0963548306007978.

9 P. Cuff, J. Ding, O. Louidor, E. Lubetzky, Y. Peres, and A. Sly. Glauber dynamics for the mean-field Potts model. *Journal of Statistical Physics*, 149(3):432–477, 2012. URL: https://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=82730695&site=eds-live&scope=site.

10 Persi Diaconis and Laurent Saloff-Coste. Comparison theorems for reversible Markov chains. *The Annals of Applied Probability*, 3:696–730, 1993.

11 Jian Ding, Eyal Lubetzky, and Yuval Peres. The mixing time evolution of Glauber dynamics for the mean-field Ising model. *Communications in Mathematical Physics*, 289(2):725–764, 2009. doi:10.1007/s00220-009-0781-9.

12 Jian Ding, Eyal Lubetzky, and Yuval Peres. Mixing time of critical Ising model on trees is polynomial in the height. *Communications in Mathematical Physics*, 295(1):161–207, 2010. doi:10.1007/s00220-009-0978-y.

13 Reza Gheissari and Eyal Lubetzky. Mixing times of critical 2D Potts models. Submitted. Available at https://arxiv.org/abs/1607.02182.

14 Catherine Greenhill. The switch Markov chain for sampling irregular graphs. In *Proceedings of the 26th Symposium on Discrete Algorithms (SODA)*, 2015.

15 OEIS Foundation Inc. The on-line encyclopedia of integer sequences, 2017. http://oeis.org/A062764.

16 Svante Janson, Dana Randall, and Joel Spencer. Random dyadic tilings of the unit square. *Random Structures and Algorithms*, 21:225–251, 2002.

17 Jeffery C. Lagarias, Joel H. Spencer, and Jade P. Vinson. Counting dyadic equipartitions of the unit square. *Discrete Mathematics*, 257(2-3):481–499, November 2002. doi:10.1016/S0012-365X(02)00508-3.

18 David A. Levin, Malwina J. Luczak, and Yuval Peres. Glauber dynamics for the mean-field Ising model: cut-off, critical power law, and metastability. *Probab. Theory Related Fields*, 146(1-2):223–265, 2010. doi:10.1007/s00440-008-0189-z.

19 David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov Chains and Mixing Times.* American Mathematical Society, Providence, RI, 2009.

20 Eyal Lubetzky and Allan Sly. Critical Ising on the square lattice mixes in polynomial time. *Communications in Mathematical Physics*, 313(3):815–836, 2012. doi:10.1007/s00220-012-1460-9.

21 Michael Luby, Dana Randall, and Alistair Sinclair. Markov chain algorithms for planar lattice structures. *SIAM Journal on Computing*, 31:167–192, 2001.

22 Fabio Martinelli. Lectures on Glauber dynamics for discrete spin models. In Pierre Bernard, editor, *Lectures on Probability Theory and Statistics: Ecole d'Eté de Probailités de Saint-Flour XXVII – 1997*, pages 93–191. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999. doi:10.1007/978-3-540-48115-7_2.

23 Leslie McShine and Prasad Tetali. On the mixing time of the triangulation walk and other Catalan structures. *DIMACS-AMS Volume on Randomization Methods in Algorithm Design*, 43:147–160, 1998.

24 N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

25 Dana Randall and Prasad Tetali. Analyzing Glauber dynamics by comparison of Markov chains. *Journal of Mathematical Physics*, 41:1598–1615, 2000.

26 Clayton Scott and Robert D. Nowak. Minimax-optimal classification with dyadic decision trees. *IEEE Transactions on Information Theory*, 52(4), 2006.

27 Alexandre Stauffer. A Lyapunov function for Glauber dynamics on lattice triangulations. *Probability Theory and Related Fields*, to appear.

A The spectral gap of the block dynamics

We now present the proof of Theorem 9, which states that there exists a positive integer k_0 such that for all $k \geq k_0$, the spectral gap $\gamma_{k,block}$ is at least 1/17.

Proof of Theorem 9. We start defining the distance between two dyadic tilings $x, y \in \Omega_k$. In order to do this, we recall the notion of *half-bisectors*. We say that a tiling x has a *left half-bisector* if the line segment from $(0, 1/2)$ to $(1/2, 1/2)$ does not intersect the interior of any dyadic rectangle. In an analogous way we can define a *right half-bisector* using the line segment from $(1/2, 1/2)$ to $(1, 1/2)$, a *top half-bisector* using the line segment from $(1/2, 1)$ to $(1/2, 1/2)$, and a *bottom half-bisector* using the line segment from $(1/2, 1/2)$ to $(1/2, 0)$. Note that if x has a horizontal bisector, then it has both a left half-bisector and a right half-bisector. However, x may have a left half-bisector but no horizontal bisector. For example, the dyadic tiling in Figure 1(a) has top, right and bottom half-bisectors, but no left half-bisector.

Now we define the distance between x and y as follows. For each of the four possible half-bisectors, let ℓ_1 be the number of such half-bisectors that are present in either x or y, but not in both of them. Also, for each of the four possible quadrants (top-left, top-right, bottom-left and bottom-right) of x and y, let ℓ_2 denote the number of such quadrants for which the rectangles in x intersecting that quadrant are not the same as the rectangles in y intersecting that quadrant. Then, introducing a parameter $b > 0$ that we will take to be sufficiently large later, we define the distance between x and y as

$$d(x, y) = b\ell_1 + \ell_2.$$

For instance, consider the two dyadic tilings in Figure 1(a,b). In this case we have $\ell_1 = 1$ due to the left half-bisector that is present in (b) but not in (a), and $\ell_2 = 3$ for top-left, top-right and bottom-left quadrants. The distance between these two tilings is then $b + 3$.

Our goal is to couple two instances of the block dynamics $\mathcal{M}_k^{block}$, one starting from a state $x \in \Omega_k$ and the other from a state $y \in \Omega_k$, such that the distance between x and y contracts after one step of the chains. More precisely, letting $\mathbb{E}_{x,y}$ denote the expectation with respected to the coupling, and if x' and y' are the dyadic tilings obtained after one step of each chain, respectively, we want to obtain a coupling and a value $\Delta > 0$ such that

$$\mathbb{E}_{x,y}[d(x', y')] \leq (1 - \Delta) d(x, y) \quad \text{for all } x, y \in \Omega_k. \tag{6}$$

Once we have the above inequality, then a result of Chen [7] (see also [19, Theorem 13.1]), implies that $\gamma_{k,block} \geq \Delta$.

We will use the following simple coupling between x' and y':

- Uniformly at random choose a tiling $\rho \in \Omega_{k-1}$.
- Uniformly at random choose *Left*, *Right*, *Top* or *Bottom*.
- Retile the choosen half (left, right, top or bottom) of x with ρ, if possible.
- Retile the choosen half (left, right, top or bottom) of y with ρ, if possible.

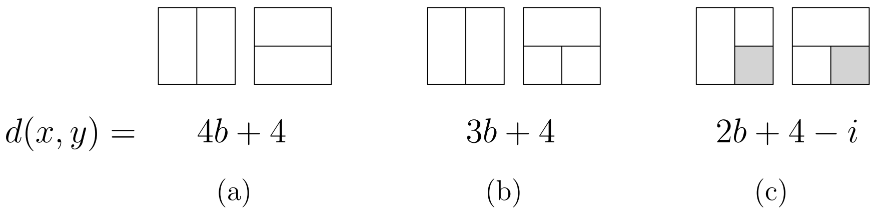


Figure 6 Possible configurations for the half-bisectors of x and y in case 1. In figure (c), $i \in \{0,1\}$ denotes how many grey quadrants are tiled identically in x and y.

For a more detailed description of the retiling step, see the definition of the transition rule of $\mathcal{M}_k^{block}$ in Section 3.1. When we update the left (resp., right) half of x and ρ contains a horizontal bisector, note that x' will contain a left (resp., right) half-bisector. Similarly, if we update the top (resp., bottom) half of x and ρ contains a vertical bisector, then x' will contain a top (resp., bottom) half-bisector. In any of these cases, we say that the retiling yields a half-bisector of x.

The remaining of the proof is devoted to showing that we can set b large enough so that (6) holds with $\Delta = \frac{1}{17}$. In order to see this, we will split into three cases, and show that (6) holds with $\Delta = \frac{1}{17}$ for each case.

Case 1: x and y have no common bisector. The maximum number of common half-bisectors of x and y in this case is two. Figure 6 illustrates the three possible configurations for the number of common half-bisectors of x and y.

Consider first that x and y have no common half-bisector, which is illustrated in Figure 6(a) and has $d(x,y) = 4b+4$. Then, whichever half (left, right, top or bottom) is chosen to be retiled, note that either x or y is actually retiled, but never both. With probability $\frac{|\Omega_{k-2}^2|}{|\Omega_{k-1}|} = f_k$ the retiling yields a half-bisector, which increases the number of common half-bisectors between x and y, and thus decreases their distance by b. Hence, using that $f_k \geq 1/2$, we have

$$\mathbb{E}_{x,y}[d(x',y')] = d(x,y) - f_k b \leq 4b + 4 - \frac{b}{2} < \left(1 - \frac{1}{17}\right)(4b+4),$$

where the last step is true by setting b large enough (in this case, $b \geq 1$ suffices).

Now consider that x and y have one common half-bisector, and use Figure 6(b) as a reference, with x being the left tiling and y being the right tiling. We have $d(x,y) = 3b+4$. If we retile the left or right halves, so only x gets retiled, and the retiling yields a half-bisector, then the number of common half-bisectors of x and y decreases by 1. A similar behavior happens if we retile the top half. However, if we retile the bottom half, and the retiling does not yield a half-bisector, then the number of common half-bisectors decreases by 1. Hence, using that $f_k \geq 1/2$, we obtain

$$\mathbb{E}_{x,y}[d(x',y')] \leq d(x,y) - \frac{3f_k b}{4} + \frac{(1-f_k)b}{4} \leq 3b + 4 - \frac{b}{4} < \left(1 - \frac{1}{17}\right)(3b+4),$$

where the last step is true by setting b large enough (in this case, $b \geq 4$ suffices).

Finally, suppose x and y have two common half-bisectors, as illustrated in Figure 6(c), where they may or may not be tiled the same in the quadrant bounded by these common half-bisectors. In this case $d(x,y) = 2b + 4 - i$, where $i = 1$ if they agree on this quadrant and $i = 0$ otherwise. Retiling the left and top halves can yield a new common half-bisector,

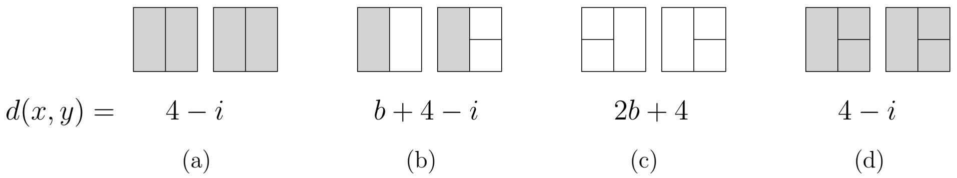


Figure 7 Possible configurations for the half-bisectors of x and y in case 2. The value of $i \in \{0, 1, 2, 3\}$ denotes the number of grey quadrants which is tiled identically in x and y.

while retiling the right and bottom halves may remove a common half-bisector. Moreover, if $i = 1$ and we retile the right or bottom halves, the tilings of the bottom-right quadrant of x and of y may become different, increasing the distance between x and y by 1. Putting these together, we have

$$\begin{aligned}\mathbb{E}_{x,y}[d(x',y')] &\le d(x,y) - \frac{2f_k b}{4} + \frac{2(1-f_k)b}{4} + i\frac{2}{4} \\ &\le 2b + 4 - \frac{i}{2} - \frac{(2f_k - 1)b}{2} = \frac{(5-2f_k)b}{2} + 4 - \frac{i}{2}.\end{aligned}$$

Since $f_k \to \frac{\sqrt{5}-1}{2}$ as $k \to \infty$, the right-hand side above goes to $\left(\frac{6-\sqrt{5}}{2}\right) b + 4 - \frac{i}{2}$. In particular, for $k \ge 10$, the coefficient of b above satisfies $\frac{5-2f_k}{2} < 2\left(1 - \frac{1}{17}\right) - 0.0002$, and so we can set b large enough so that $\mathbb{E}_{x,y}[d(x',y')] \le \left(1 - \frac{1}{17}\right)(2b + 4 - i)$. We note that as $\frac{6-\sqrt{5}}{2} > 2\left(1 - \frac{1}{16}\right)$, this particular coupling and distance metric cannot be used to show the spectral gap is at least 1/16. This concludes the first case.

Case 2: x and y have a common bisector, but neither x nor y has both bisectors. Without loss of generality we assume x and y both have a vertical bisector and neither has a horizontal bisector. Each of x and y has at least 2 and at most 3 half-bisectors. Figure 7 illustrates the four possible configurations for the number of half-bisectors of x and y; the shaded quadrants are those where x and y could have the same tiling.

In all the situations of Figure 7, if we retile the left or right halves, then we match up the configuration of x and y in that half. In particular, if x and y don't agree on the presence of left half-bisector, then they also do not have the same tiling of the top left or bottom left quadrants, so the decrease in distance due to a retiling of the left half, a move that occurs with probability 1/4, is $(b + 2)$. If x and y agree on the presence of a left half-bisector and have the same tiling on $i' \in \{0, 1, 2\}$ of the two left quadrants, then the decrease in distance due to a retiling of the left half is $(2 - i')$. The same holds for right half-bisectors and retilings of the right half. As there are no moves of the coupling that can increase the distance between x and y, it can be shown that in all of the cases shown in Figure 7 the distance decreases by 1/4 in expectation. Hence,

$$\mathbb{E}_{x,y}[d(x',y')] \le d(x,y) - \frac{d(x,y)}{4} \le \left(1 - \frac{1}{17}\right) d(x,y),$$

which concludes the second case.

Case 3: y has both vertical and horizontal bisectors. Here there are three situations, depending on whether x has two, three or four half-bisectors; see Figure 8.

In the situation of Figure 8(a), if the left or right halves are retiled, then we match up x and y in that half, decreasing the distance by $b + 2$. But if we retile the top or bottom halves,

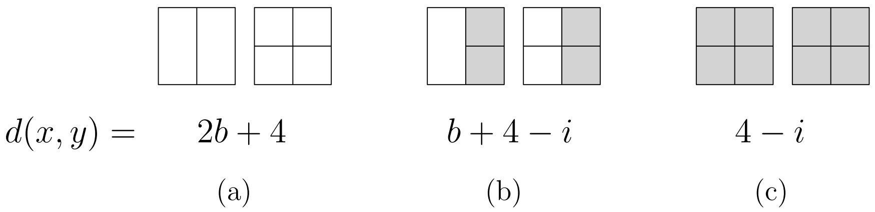


Figure 8 Possible configurations for the half-bisectors of x and y in case 3. The value of $i \in \{0,1,2,3\}$ denotes the number of grey quadrants which is tiled identically in x and y.

then we may increase the distance by b if the retiling does not yield a half-bisector. Hence,

$$\mathbb{E}_{x,y}[d(x',y')] \leq d(x,y) - \frac{2(b+2)}{4} + \frac{2(1-f_k)b}{4} = \frac{(4-f_k)b}{2} + 3.$$

Since $\frac{4-f_k}{2} \to \frac{9-\sqrt{5}}{4} < \left(1-\frac{1}{17}\right)2$, the right-hand side above is smaller than $\left(1-\frac{1}{17}\right)(2b+4)$ when k and b are large enough. A similar situation occurs in Figure 8(b), but the distance increases a bit more when the top or bottom half is retiled as quadrants that were equal in x and y may become different. In this case, we have

$$\mathbb{E}_{x,y}[d(x',y')] \leq d(x,y) - \frac{(b+4-i)}{4} + \frac{2(1-f_k)b}{4} + \frac{2}{4} = \frac{(5-2f_k)b}{4} + \frac{6-i}{4}.$$

Since $\frac{5-2f_k}{4} \to \frac{6-\sqrt{5}}{4} < \left(1-\frac{1}{17}\right)$, the right-hand side above is smaller than $\left(1-\frac{1}{17}\right)(b+4-i)$ when k and b are large enough; as in Case 1, we obtain a contraction by a factor of $1-\frac{1}{17}$ but not by $1-\frac{1}{16}$. Finally, for the situation in Figure 8(c), regardless of which half we choose to retile, the distance will not increase; if we choose a half containing a quadrant on which x and y differ, the distance will decrease. Each quadrant on which x and y differ is contained in two halves and thus is retiled so that x and y agree there with probability 1/2. That is,

$$\mathbb{E}_{x,y}[d(x',y')] \leq d(x,y) - \frac{d(x,y)}{2} \leq \left(1-\frac{1}{17}\right)d(x,y).$$

This concludes the third case. We have shown that for all possible tilings x and y, it holds that $\mathbb{E}_{x,y}[d(x',y')] \leq \left(1-\frac{1}{17}\right)d(x,y)$. This implies $\gamma_{k,block} \geq \frac{1}{17}$ for all k sufficiently large, as desired. ◀

B Omitted Proofs

Here we include proofs of some basic facts about dyadic tilings and their structure that were omitted in Section 2 due to space constraints. Recall that f_k is the fraction of all dyadic tilings in Ω_k with a left half-bisector.

▶ **Lemma 4.** *For all $k \geq 3$, $f_k = \frac{1}{2-f_{k-1}^2}$.*

Proof. This follows from the recurrence for A_k given in Proposition 3:

$$f_k = \frac{A_{k-2}^2}{A_{k-1}} = \frac{A_{k-2}^2}{2A_{k-2}^2 - A_{k-3}^4} = \frac{1}{2-\frac{A_{k-3}^4}{A_{k-2}^2}} = \frac{1}{2-f_{k-1}^2}.$$ ◀

▶ **Lemma 5.** *The sequence $\{f_k\}_{k=2}^{\infty}$ is strictly increasing and bounded above by $(\sqrt{5}-1)/2$. Furthermore, $\lim_{k\to\infty} f_k = (\sqrt{5}-1)/2$.*

Proof. Note $f_2 = 0.5 < (\sqrt{5}-1)/2$. Suppose by induction that $f_{k-1} < \frac{\sqrt{5}-1}{2}$. Then

$$f_k = \frac{1}{2 - f_{k-1}^2} < \frac{1}{2 - \left(\frac{\sqrt{5}-1}{2}\right)^2} = \frac{4}{8 - (6 - 2\sqrt{5})} = \frac{4}{2 + 2\sqrt{5}} = \frac{2}{1 + \sqrt{5}} = \frac{\sqrt{5}-1}{2}.$$

To show $f_k < f_{k+1}$ for all $k \geq 3$, it suffices to show $x < 1/(2 - x^2)$ for all $x \in \big[0.5, (\sqrt{5}-1)/2\big)$. This is equivalent to showing the polynomial $x^3 - 2x + 1$ is positive in that range. Factoring shows this polynomial has roots at 1, $(\sqrt{5}-1)/2$, and $-(\sqrt{5}+1)/2$, and is positive in the range $\big(-(\sqrt{5}+1)/2, (\sqrt{5}-1)/2\big)$. This implies $f_k < f_{k+1}$, so the sequence is strictly increasing.

The sequence $\{f_k\}_{k=2}^{\infty}$ is bounded and monotone, so it must converge to some limit β. To find β, we consider the function $g(x) = 1/(2 - x^2)$, which is the recurrence for the f_k. This function is continuous away from $\sqrt{2}$ and $-\sqrt{2}$, and thus certainly is continuous on $\big[0.5, (\sqrt{5}-1)/2\big]$, the range of possible values for the f_k and their limit β. This continuity implies

$$g(\beta) = g\left(\lim_{k\to\infty} f_k\right) = \lim_{k\to\infty} g(f_k) = \lim_{k\to\infty} f_{k+1} = \beta.$$

Thus the limit β is necessarily a fixed point of $g(x)$. The fixed points of $g(x)$ are exactly the three roots of $x^3 - 2x + 1$ found above, and the only one in $\big[0.5, (\sqrt{5}-1)/2\big]$ is $(\sqrt{5}-1)/2$. We conclude $\lim_{k\to\infty} f_k = (\sqrt{5}-1)/2$, as desired. ◀

Agnostic Learning from Tolerant Natural Proofs

Marco L. Carmosino[1], Russell Impagliazzo[2], Valentine Kabanets[3], and Antonina Kolokolova[4]

1 **Department of Computer Science, University of California San Diego, La Jolla, CA, USA**
mcarmosi@cs.ucsd.edu

2 **Department of Computer Science, University of California San Diego, La Jolla, CA, USA**
russell@cs.ucsd.edu

3 **School of Computing Science, Simon Fraser University, Burnaby, BC, Canada**
kabanets@cs.sfu.ca

4 **Department of Computer Science, Memorial University of Newfoundland, St. John's, NL, Canada**
kol@mun.ca

Abstract

We generalize the "learning algorithms from natural properties" framework of [4] to get *agnostic* learning algorithms from natural properties with extra features. We show that if a natural property (in the sense of Razborov and Rudich [28]) is useful also against functions that are *close* to the class of "easy" functions, rather than just against "easy" functions, then it can be used to get an agnostic learning algorithm over the uniform distribution with membership queries.

- For $\mathsf{AC}^0[q]$, any prime q (constant-depth circuits of polynomial size, with AND, OR, NOT, and MOD_q gates of unbounded fanin), which happens to have a natural property with the requisite extra feature by [27, 31, 28], we obtain the first agnostic learning algorithm for $\mathsf{AC}^0[q]$, for every prime q. Our algorithm runs in randomized quasi-polynomial time, uses membership queries, and outputs a circuit for a given boolean function $f\colon \{0,1\}^n \to \{0,1\}$ that agrees with f on all but at most $(\mathsf{poly}\log n)\cdot\mathsf{opt}$ fraction of inputs, where opt is the relative distance between f and the closest function h in the class $\mathsf{AC}^0[q]$.
- For the ideal case, a natural proof of strongly exponential correlation circuit lower bounds against a circuit class $\mathcal{C}$ containing $\mathsf{AC}^0[2]$ (i.e., circuits of size $\exp(\Omega(n))$ cannot compute some n-variate function even with $\exp(-\Omega(n))$ advantage over random guessing) would yield a polynomial-time query agnostic learning algorithm for $\mathcal{C}$ with the approximation error $O(\mathsf{opt})$.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases agnostic learning, natural proofs, circuit lower bounds, meta-algorithms, $\mathsf{AC}^0[q]$, Nisan-Wigderson generator

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.35

1 Introduction

Recently many new connections have been discovered between the two complementary domains: proving circuit lower bounds and designing meta-algorithms for the corresponding circuit classes (see, e.g., [30, 33, 34, 15, 16, 5, 4]). In particular, [4] shows that a natural property (in the sense of Razborov and Rudich [28]) for a (sufficiently powerful) circuit class Λ yields an efficient PAC learning algorithm for the same circuit class, under the uniform

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 35; pp. 35:1–35:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

distribution, with membership queries; this approach led to a first learning algorithm for the class $\mathsf{AC}^0[q]$ (of constant-depth circuits with AND, OR, NOT, and modulo q gates), for every prime q.

The "learning algorithms from natural proofs" technique of [4] applies only to *realizable case* learning: if a function f is computed exactly by an appropriate circuit class Λ for which there is a natural proof of a circuit lower bound, then we can learn f using membership queries in time dependent on the strength of the circuit lower bound. A more realistic learning model is *agnostic learning*, where we select some "touchstone" class Λ and attempt to find a hypothesis that isn't "too far off" from the best Λ-approximation to the target function.

We show that, even in this agnostic setting, we can (somewhat generically) obtain learning algorithms from natural proofs. We instantiate this framework to give the first membership-query agnostic learning algorithm over the uniform distribution for $\mathsf{AC}^0[q]$, the class of constant-depth circuits of polynomial-size with unbounded fanin AND, OR, NOT, and MOD_q gates. Previously, only the case of AC^0 circuits was known (albeit for an agnostic algorithm without membership queries, and with better approximation error) [19] (based on the LMN algorithm of [23]).

► **Theorem 1** ($\mathsf{AC}^0[q]$ agnostic learning)**.** *Let q be any prime. There is a randomized quasi-polynomial-time algorithm such that, given oracle access to a function $f\colon \{0,1\}^n \to \{0,1\}$ that agrees with some unknown function in $\mathsf{AC}^0[q]$ on at least $1-\beta$ fraction of inputs (for some non-negligible $\beta > 0$), the algorithm outputs a circuit that computes f on all but at most $\mathsf{poly}(\log n) \cdot \beta$ fraction of inputs.*

As an interesting special case, we get a quasipolynomial-time agnostic learning algorithm for n-variate polynomials over $\mathbf{GF}(q)$ of low degree (say, at most $\mathsf{poly}(\log n)$), for prime $q \geq 2$ (as every polynomial of degree d is computable by an $\mathsf{AC}^0[q]$ circuit of size $O(n^d)$). Before our result, no such learning algorithm for polynomials was known.

For an algorithm with error $c(n) \cdot \beta$, for some function c, we call the factor $c(n)$ the *weakness* parameter of the learning algorithm. It is desirable to have $c(n) = 1$. Our algorithm for $\mathsf{AC}^0[q]$ above has weakness $\mathsf{poly}(\log n)$. In general, we have a trade-off between the quality of a natural property for the circuit class, and the quality of the resulting agnostic learning algorithm for the same class. For simplicity, we state here just the result for the best-case scenario; see Theorem 11 below for the fully general statement.

► **Theorem 2** (Ideal-case trade-off)**.** *Suppose there is a natural property for a circuit class $\mathcal{C} \supseteq \mathsf{AC}^0[2]$ that is useful against functions that agree on $1/2 + \exp(-\Omega(n))$ of inputs with some function of $\mathcal{C}$-circuit complexity $\exp(\Omega(n))$. Then, for some constant $c > 0$, there is a polynomial-time query agnostic learning algorithm for $\mathcal{C}$ with weakness c.*

Theorem 2 yields a "search-to-decision" reduction for a version of the Minimal Circuit Size Problem (MCSP). Define the Minimal Approximate Circuit Size Problem (MACSP) as follows: Given a truth table of an n-variate boolean function f, and parameters $s \in \mathbb{N}$ and $\delta \in [0,1]$, decide if there exists a boolean circuit C of size at most s that agrees with f on all but at most δ fraction of inputs. (MCSP is a special case of MACSP for $\delta = 0$.) Clearly, if MACSP is easy (say, in P), then, for a given size bound s (our "budget"), we can determine the best approximation parameter δ for every given truth table of a boolean function f. But, since MACSP is an ideal-case tolerant natural property for general circuits, we get by Theorem 2 that a polynomial-time algorithm for MACSP would yield a polynomial-time algorithm to actually find a circuit of size $\mathsf{poly}(s)$, with an approximation guarantee $O(\delta)$.[1]

[1] In [4], a similar "search-to-decision" reduction was given for MCSP: if a given boolean function f is

Another way to interpret Theorem 2 is as follows. If MACSP is in P, then, given oracle access to a boolean function f, and a budget $s \in \mathbb{N}$, we can learn, in polynomial time, a circuit of size $\mathsf{poly}(s)$ that agrees with f on all but at most $O(\delta)$ fraction of inputs, where δ is the error of the best size s circuit for f. That is, we can learn essentially the best possible circuit for f, given our budget s on the circuit size.

1.1 Our approach

The key observation in adapting to the agnostic setting is that many natural properties contain even more useful distinguishers than required for realizable-case learning. As defined by [28], the distinguisher from a natural property rejects truth tables that are exactly computed by Λ-circuits. But existing natural properties give us something even stronger: they reject truth tables which are just *close* to those computed by Λ-circuits. Using this observation and the same “play to lose” distinguisher-to-predictor reduction as in [4], we obtain agnostic learning algorithms from such natural properties.

More precisely, we show that if a natural property for a circuit class Λ (containing $\mathsf{AC}^0[q]$) is *tolerant* in the sense that it distinguishes from random the truth tables of functions “close” to the class Λ (of “large” circuit complexity), then it can be used to get an agnostic membership-query algorithm for learning Λ. We argue that such a tolerant natural property exists for $\mathsf{AC}^0[q]$ [27, 31, 28], which is then used to prove our Theorem 1. For $\mathsf{AC}^0[2]$, we need to dig inside the arguments of [27], and show that his original circuit lower bound proof does yield a certain tolerant natural property. For $\mathsf{AC}^0[q]$, for prime $q > 2$, we actually need to re-do the “natural proof” argument of [28] by adapting it to the case of $\mathbf{GF}(q)$-valued functions (rather than boolean functions). Not only does it allow us to get tolerant natural properties for $\mathsf{AC}^0[q]$, but also simplifies and streamlines the analysis in [4] of the learning algorithm for $\mathsf{AC}^0[q]$.

By definition, tolerant natural properties can be used for proving *average-case* circuit lower bounds (as opposed to the worst-case circuit lower bounds implied by standard natural properties). Thus the main message of the present paper can be summarized as follows:

> Natural proofs of *average-case* circuit lower bounds imply *agnostic* learning algorithms!

In contrast, the main result of [4] says that natural proofs of worst-case circuit lower bounds imply standard (non-agnostic) learning algorithms.

1.2 Our techniques

We build upon the framework of [4] who use a natural property for a given circuit class Λ in order to devise a learning algorithm for the same class. Recall that a natural property (in the sense of [28]) is an efficient algorithm that tells apart truth tables of functions in the class Λ (of some “large” circuit complexity u) from those of random functions. To learn a function $f \in \Lambda$, for some circuit class Λ that has an associated natural property, the idea is to apply (as only a thought experiment!) an appropriate “function generator” that maps f to a family of functions all of which are “easy” (of small Λ circuit complexity) and so will be rejected by

exactly computable by a polynomial-size circuit, then one can find a polynomial-size circuit *approximately* computing f, given a polynomial-time algorithm for MCSP. In contrast, here we say that if f can be non-trivially *approximated* by a polynomial-size circuit, we can find another polynomial-size circuit that achieves the *same approximation error* up to a constant factor, given a polynomial-time algorithm for MACSP.

the natural property for the class. Thus an efficient algorithm from the natural property acts as a distinguisher "breaking" the function generator. If the function generator has an "efficient reconstruction" property, meaning that a distinguisher for the generator can be used to build a small circuit approximately computing the original function f, we get a learning algorithm for f. Thus, the actual learning algorithm is using the natural property algorithm as a distinguisher, and applies the efficient reconstruction procedure (associated with the given function generator) to build a circuit approximating f. Usually, such a reconstruction procedure requires oracle access to the function generator; if, however, the function generator is "local" in the sense that such oracle access to the generator can be efficiently reduced to oracle access to the original function f, one gets a query learning algorithm for the concept class Λ.

To adapt this approach to the case of agnostic learning, where a function f to be learned is not in the class Λ, but rather just somewhat close to the class, we need to satisfy the following requirements:

1. the outputs of the function generator applied to f must be close to the class Λ (of some circuit size u), and
2. the natural property for Λ must reject not only functions in Λ (of size u), but also functions that are close to those.

We call natural properties satisfying condition (2) above *tolerant*. We say that a natural property has ρ-tolerant u-usefulness for the circuit class Λ if it rejects all truth tables of functions that agree with some function in $\Lambda[u]$ (computable by a Λ circuit of size u) on all but at most ρ fraction of inputs. We show that the natural property for the circuit class $\mathsf{AC}^0[2]$ from [27] is in fact ρ-tolerant, for some small but nontrivial $\rho > 0$, and with large (weakly-exponential) usefulness u.

With tolerant natural properties in hand, we turn to requirement (1) above: getting the truth tables output by the function generator on a given function f to be close to those from the circuit class $\Lambda[u]$. We need to take a closer look at the function generator used in [4]. It comprises two components: (1) amplification, and (2) Nisan-Wigderson (NW) generator [25] applied to the amplified version $\mathsf{Amp}(f)$ of the function f. The purpose of the amplification component is to "error-correct" f so that even a circuit that computes $\mathsf{Amp}(f)$ with small advantage over random guessing can be used to construct a circuit that computes f almost everywhere. The NW generator applied to $\mathsf{Amp}(f)$ has the properties required of the function generator: locality and efficient reconstruction.

In our case, suppose that f agrees with some function $h \in \Lambda$ on a large fraction of inputs. Once we apply amplification to both f and h, we get $\mathsf{Amp}(f)$ and $\mathsf{Amp}(h)$ that are pushed further apart (as one would expect when using error-correcting codes). In order to keep the amplified functions close to each other, we will tone down the amplification procedure, which will adversely affect the approximation error of our learning algorithm, but the error can still be kept relatively small.

Next we need to ensure that the NW generator when applied to $\mathsf{Amp}(f)$ generates a family of functions such that most of them are sufficiently close to the family generated on $\mathsf{Amp}(h)$. In other words, we would like the generator to almost preserve the relative distance between the functions it is applied to. This can be achieved as follows. First, we observe that the definition of the NW generator guarantees that on a random seed z, the functions generated for $\mathsf{Amp}(f)$ and $\mathsf{Amp}(h)$ have the expected distance (over random z) equal to the actual distance between $\mathsf{Amp}(f)$ and $\mathsf{Amp}(h)$. Thus we have distance preservation in expectation. To make it concentrated around the expectation, we modify the NW construction by adding a pairwise-independent generator inside the NW construction. This ensures that the truth tables output by the modified NW generator are evaluations of $\mathsf{Amp}(f)$ (or $\mathsf{Amp}(h)$) on a sequence of pairwise independent inputs. The required concentration then follows by the Chebyshev bound. (A similar modification of the NW generator was done in [17], where an expander-walk generator was used for even better concentration; we use a simple pairwise generator as it can be easily implemented in AC^0[2], and it provides sufficient concentration for our purposes.)

1.3 Related work

The concept of *agnostic* learning was introduced by Kearns et al. [20], where it was also shown that piecewise linear functions are agnostically learnable. Agnostic learning is also known for certain geometric patterns [10], and restricted neural networks [21]. More results are known for the restricted versions of agnostic learning, for instance, when the distribution over examples is uniform. The class of AC^0 functions was shown to be (weakly) agnostically learnable under the uniform distribution by [20]. It was later shown by [19] that the well-known LMN learning algorithm of [23] achieves a constant-factor approximation of the optimal error (improved to the constant factor 2 in [18]), and that a modification of the algorithm (using L_1 regression) achieves the optimal error; the runtime of the algorithm is quasipolynomial. In fact, the result of [19] is generic in the following sense: any concept class of functions with certain "Fourier concentration" (as is the case, e.g., for AC^0 functions by the results of [23]) admits an agnostic learning algorithm under the uniform distribution, with an optimal error, whose runtime depends on the strength of the Fourier concentration for the concept class.

In distribution-independent setting, allowing membership queries does not give extra power to agnostic learning, yet membership queries can help when the distribution is uniform [6]. In particular, under the uniform distribution, Gopalan, Kalai and Klivans [12] and Feldman [7] give polynomial-time agnostic learning algorithms with membership queries for decision trees.

Agnostic learning of parities is closely related to the well-studied problem of learning noisy parities, which has a number of applications beyond learning theory, from decoding random linear codes to cryptography[2, 9, 1, 24, 26].

Under the uniform distribution, agnostic learning of parities (that is, learning parities with adversarial noise) reduces to learning parities with random noise [8]. Blum, Kalai and Wasserman [3] give an algorithm that properly learns length k parities with random noise under uniform distribution in time and sample size $\mathsf{poly}((1/(1-2\eta))^{2^a}, 2^b)$, where $\eta < 1/2$ is the noise probability, and $ab \geq k$. This is in contrast to the NP-hardness of properly learning noisy parities under arbitrary distributions, which follows from [13]. Later, Lyubashevsky [24] improved query complexity of the [3] algorithm to $n^{1+\epsilon}$, at the expense of bringing the running time up to $2^{O(n/\log\log n)}$, for $\eta < 1/2 - 2^{-(\log n)^\delta}$ for a constant δ. A corollary of the latter result is a subexponential algorithm for decoding $n \times n^{1+\epsilon}$ random binary linear codes, in the random noise setting.

Regev [29] considered an extension of learning parity with noise to mod p, which he called LWE (learning with error). He has shown that an efficient solution to LWE (for some range of parameters) implies an efficient quantum approximation of two variants of the shortest vector problem (GapSVP and the shortest independent vectors problem) and presented a public-key cryptosystem based on its hardness.

Remainder of the paper

We start with some basic definitions in Section 2. In Section 3, we prove our main result, Theorem 1, by instantiating the "agnostic learning from tolerant natural properties" framework to the case of $\mathsf{AC}^0[q]$ circuits, for any prime q. We present this framework in full generality in Section 4, where, in particular, we prove Theorem 2. In Section 5, we discuss the difficulty of removing membership queries from our agnostic learning algorithms for $\mathsf{AC}^0[2]$ (as it would have consequences for learning noisy parities). We conclude with some open questions in Section 6. The appendix contains some proofs omitted from the main body of the paper.

2 Preliminaries

For n-variate boolean functions f and g, we define the distance between them, denoted $\mathrm{DIST}(f,g)$, to be the number of inputs x where $f(x) \neq g(x)$. We denote by $\mathrm{dist}(f,g)$ the relative distance $\mathrm{DIST}(f,g)/2^n$. For a class $\mathcal{F}$ of n-variate boolean functions, and an n-variate boolean function f, we define the distance of f from the class $\mathcal{F}$, denoted $\mathrm{DIST}(f,\mathcal{F})$, as $\min_{h\in\mathcal{F}} \mathrm{DIST}(f,h)$. The relative distance of f from $\mathcal{F}$ is $\mathrm{dist}(f,\mathcal{F}) = \mathrm{DIST}(f,\mathcal{F})/2^n$.

▶ **Definition 3** (Distinguishers). Let $L\colon \mathbb{N} \to \mathbb{N}$ be a stretch function, let $0 < \epsilon < 1$ be an error bound, and let $\mathcal{G} = \{g_m\colon \{0,1\}^m \to \{0,1\}^{L(m)}\}$ be a sequence of functions. Define $\mathsf{DIS}(\mathcal{G},\epsilon)$ to be the set of all Boolean circuits D on $L(m)$-bit inputs satisfying: $\Pr_{z\in\{0,1\}^m}[D(g_m(z))] - \Pr_{y\in\{0,1\}^{L(m)}}[D(y)] > \epsilon$. We say that $D \in \mathsf{DIS}(\mathcal{G},\epsilon)$ is a *distinguisher for $\mathcal{G}$ with the distinguishing probability ϵ.*

2.1 Learning algorithms

The concept of *agnostic learning* was introduced by [20]. As in the PAC model of Valiant [32], we have a distribution over labeled examples $(x, f(x))$ for some function f, and we wish to learn f up to a small additive error over the given distribution. However, unlike in the PAC model, we don't assume that f belongs to some concept class $\mathcal{C}$, but rather that f is "close" to $\mathcal{C}$. More precisely, setting opt to be the disagreement probability between f and the best (closest) function $h \in \mathcal{C}$, the agnostic learning algorithm is supposed to output, with high probability $1-\delta$, a hypothesis that disagrees with f with probability at most $\mathsf{opt}+\epsilon$, for given $\epsilon, \delta \in [0,1]$. If the underlying distribution over examples is uniform, we say that the concept class $\mathcal{C}$ is agnostically learnable under the uniform distribution.

In the special case where we allow membership oracle, i.e., our learning algorithm has oracle access to the function f it is trying to learn, we call it a (membership) query agnostic learning algorithm. If, in addition, the hypothesis error is measure under the uniform distribution, we call it a query agnostic learning algorithm under the uniform distribution.

The learning algorithms considered in our paper are query algorithms under the uniform distribution. However, they don't achieve the ideal error $\mathsf{opt}+\epsilon$. Rather, we get the error of the form $c(n)\cdot\mathsf{opt}$, for some function c, which we call the *weakness* parameter of the agnostic learning algorithm; we also assume that opt is non-negligible and so we can drop the additive error ϵ to simplify the notation. For example, in the case of $\mathcal{C} = \mathsf{AC}^0[2]$, our learning algorithm has weakness $\mathsf{poly}(\log n)$.

2.2 Tolerant natural properties

We extend the definition of a natural property [28] to the case of a tolerant one, which intuitively says that not only all "easy" functions are rejected by the property, but also all functions "sufficiently close" to the "easy ones" are rejected. Such tolerant properties yield not just worst-case, but also average-case circuit lower bounds.

Let F_n be the collection of all Boolean functions on n variables. Λ and Γ denote complexity classes. A combinatorial property is a sequence of subsets of F_n for each n.

▶ **Definition 4** (Tolerant Natural Property). A combinatorial property $\{R_n\}_{n\geq 0}$ is Γ-natural with density δ and τ-tolerant u-usefulness, for some functions $\delta, \tau : \mathbb{N} \to [0,1]$ and $u : \mathbb{N} \to \mathbb{N}$, if it satisfies the following conditions:

Γ-Constructivity: Given the truth table of f_n, a Γ-algorithm decides if $f_n \in R_n$.

δ-Largeness: $|R_n| \geq \delta(n) \cdot |F_n|$.

τ-Tolerant u-Usefulness: For all $f_n \in F_n$ (for large n), if $\mathrm{dist}(f_n, \Lambda[u(n)]) \leq \tau(n)$, then $f_n \notin R_n$.

The standard natural property [28] is 0-tolerant in our language. For a number of complexity classes, including $\mathsf{AC}^0[q]$ for primes q, 0-tolerant natural properties were given in [28]. We prove that the natural property of [27] has in fact $(1/n^3)$-tolerant usefulness against d-depth $\mathsf{AC}^0[2]$ circuits of size $\exp(\Omega(n^{1/(2d)}))$; see Section A of the appendix for the proof of the following.

▶ **Lemma 5** (Tolerant natural property for $\mathsf{AC}^0[2]$). *There is a* P*-natural property* $\{R_n\}_{n\geq 0}$ *with largeness* $1/2$*, and* $(1/n^3)$*-tolerant* $\exp(\Omega(n^{1/(2d)}))$*-usefulness against* d*-depth* $\mathsf{AC}^0[2]$ *circuits.*

3 Agnostic learning from tolerant natural properties for $\mathsf{AC}^0[2]$

3.1 The CIKK framework

Recall the way non-agnostic learning algorithms follow from natural properties in the framework of [4]. Suppose we want to learn a function f in some circuit class Λ; for simplicity, assume f has polynomial-size circuits of type Λ.

As a thought experiment, imagine the following transformations applied to f. First, we *amplify* f, getting a new function $F = \textsc{Amp}(f)$, on polynomially larger inputs, with the property:

> If we are given a small circuit computing F on at least $1/2 + \epsilon$ fraction of inputs, then we can construct a circuit computing the original function f on at least $1 - 1/\mathsf{poly}(n)$ fraction of inputs, in randomized time $\mathsf{poly}(n, 1/\epsilon)$, using membership queries to f.

Then F is used as a "hard function" for the NW generator G. For each seed z of the NW generator, we view the output binary string $G(z)$ of length L as the truth table of an ℓ-variate boolean function, for $\ell = \log L$. The crucial observation in [4] is that the circuit complexity of this ℓ-variate boolean function is polynomial in the circuit size of the original function f, which is $\mathsf{poly}(n)$.

We need to express this circuit complexity $\mathsf{poly}(n)$ as the function of the input size ℓ. Note that if the stretch L is small, for example, if $L = \mathsf{poly}(n)$, then $\ell = O(\log n)$, and so the ℓ-variate function (whose truth table is) output by $G(z)$ has circuit complexity exponential in its input size ℓ. Thus, to reduce the circuit complexity of the function output by $G(z)$, we need to increase the stretch L of the NW generator. For example, by taking $L = \exp(\mathsf{poly}\log n)$, we can ensure that the circuit complexity of $G(z)$ (for each seed z) is only weakly exponential in the input size ℓ.

The point of using the NW generator to produce truth tables of relatively easy functions $G(z)$ is that we assumed the existence of an efficient natural property (with sufficient usefulness) which will accept many random truth tables, but will reject all truth tables of easy functions. In other words, this natural property provides an efficient (polynomial-time) algorithm that *distinguishes* the outputs of the NW generator G from truly random strings.

But then, the analysis of the NW generator construction implies that we get from this distinguisher a new algorithm that computes F (the function upon which the NW generator was based) on at least $1/2 + \Omega(1/L)$ fraction of inputs; where the reconstruction algorithm requires membership oracle for f. The latter implies (by the aforementioned properties of $F = \textsc{Amp}(f)$) that we can construct a circuit computing f on almost all inputs, in time $\mathsf{poly}(n, L)$ (again, using membership oracle for f). Thus we get a learning algorithm from the natural property, using the efficient reconstruction algorithm for the NW generator and the amplification procedure.

For example, using natural properties against $\mathsf{AC}^0[2]$ that are useful against circuits of weakly-exponential size [28], the above framework yields a learning algorithm, with membership queries, for functions computable by polynomial-size $\mathsf{AC}^0[2]$ circuits, running in quasipolynomial time.

3.2 Extension to the agnostic learning case

We wish to apply the same framework to the task of agnostic learning. Suppose we wish to learn a function f which is only somewhat close to a function h in some circuit class Λ (of polynomial-size circuits). Suppose that $\text{dist}(f, \Lambda) \leq \beta$, and that $h \in \Lambda$ is the closest function to f. Assume we are given a membership oracle for f.

To apply the [4] approach to learn f, we need to ensure the following:

> For most seeds z, the function $G(z)$ (for the NW generator based on $F = \textsc{Amp}(f)$) is rejected by the appropriate natural property for our circuit class Λ.

If so, then we have a distinguisher for the NW generator based on F, and, as before, can efficiently construct a circuit for computing f almost everywhere.

As f is not in the class Λ, but rather just close to it, the best we can hope for is that the amplified function $F = \textsc{Amp}(f)$ is also somewhat close to Λ, and that the outputs of the NW generator $G(z)$ based on F are also somewhat close to the class Λ (of larger circuit size). If we can guarantee that (most of) the strings $G(z)$ are at the relative distance at most τ from $\Lambda[u]$, then our natural property with τ-tolerant u-usefulness will be a distinguisher for the NW generator, and we can reconstruct a circuit approximately computing f.

We need to balance the opposing constraints. On the one hand, to keep $F = \textsc{Amp}(f)$ close to Λ, we cannot amplify f too much, as the amplification, like an error-correcting encoding, pushes the originally close functions far apart. On the other hand, the stronger the amplification applied to f, the smaller the approximation error we get from a circuit for f constructed by the learning algorithm. As we are restricted by the tolerance parameter τ of our natural property, we are forced to keep the amplification relatively weak, which in turn implies a weak approximation error for the learned circuit for f.

Suppose that $f \colon \{0,1\}^n \to \{0,1\}$ is at the relative distance β from some n-variate function $h \in \Lambda[\mathsf{poly}]$. We will fine-tune the amplification procedure of [4] so that $F = \textsc{Amp}(f)$ and $H = \textsc{Amp}(h)$ are at the relative distance at most $\mu(n)$, for some $\mu : \mathbb{N} \to [0,1]$ to be determined. Then we need to ensure that the outputs of the NW generator on F and on H, for most random seeds z, produce truth tables of length L that are at the relative distance at most $\tau(\ell)$ from each other, where $\ell = \log L$ is the input size of such a function output by $G(z)$.

To ensure that the NW generator based on close functions F and H produces strings that are close (for most seeds z), we modify the NW generator by adding a pairwise-independent generator as an extra component. (Similar modification to the NW generator, using an expander-walk generator, was done in [17], for a different purpose.) We will show that such a modified NW generator, when run on functions F and H that are at the relative distance $\mu(n)$ from each other, indeed outputs, for most seeds z, strings $G^F(z)$ and $G^H(z)$ of length L each, which are at the relative distance at most $2\mu(n)$ from each other. Expressing $2\mu(n)$ as a function of the input length $\ell = \log L$, we get an upper bound on the relative distance between $G^F(z)$ and $\Lambda[u]$ (as $G^H(z) \in \Lambda[u]$ by our assumption that $h \in \Lambda[\mathsf{poly}]$), for most seeds z. Here we choose the stretch L long enough so that the circuit complexity of the functions $G^H(z)$ is at most u, where u is usefulness of our natural property. For example, for $\mathsf{AC}^0[2]$, we have usefulness against weakly-exponential circuit size $\exp(n^{1/(2d)})$ for depth d circuits, and so we can make L to be quasi-polynomial, $\exp(\mathsf{poly}\log n)$.

3.3 Outline of the general method

In converting a tolerant natural property to an agnostic learning algorithm, we go through the following steps, mostly analogous to the steps in [4].

Initial assumptions. We start with access via membership queries to a Boolean function f. We are promised that there is a function $h \in C$ so that $\mathrm{dist}(f, h) \le \beta$, for some parameter β. We do not have any access to h, but can refer to it in the analysis.

Amplification. The first step is to perform an *amplification construction*, $\mathsf{Amp}(f)$, to obtain a function F. Similarly, we can (conceptually) apply $\mathsf{Amp}(h)$ to obtain a function H. We need the following properties:

1. We can simulate membership queries to F via membership queries to f
2. $H \in C$
3. We can bound $\mathrm{dist}(F, H)$ away from $1/2$. The exact bound we will require will depend on the tolerance of the natural property.

Pseudo-random Function Generator. We next convert F to a *pseudo-random function generator*, $G_s^F(I)$ (and, conceptually, convert H into $G_s^H(I)$. For each seed s, G_s^F is a Boolean function on ℓ bits, producing a truth table of size $L = 2^\ell$. We call L the *stretch* of the generator. We need the following properties:

1. Given s, the truth table for G_s^F can be computed via membership queries to F (and hence, f).
2. For each s, $G_s^H(I)$ has small C circuit complexity (as a function of ℓ bit input I)
3. With good probability over s, $\mathrm{dist}(G_s^F, G_s^H)$ is small

Again, the exact quantitative requirements will depend on the quality of the tolerant natural property. The stronger the circuit lower bound the property is useful against, the smaller we can make the stretch and so the larger the relative circuit complexity of G_s^H in (2) can be. The more tolerant the property is, the larger the allowed distance in (3) can be. The greater the density, the smaller the probability over seeds of small distance between G_s^F and G_s^H in (3) can be.

Apply tolerant natural property to get a distinguisher. Now we use the tolerant natural property as a *distinguisher*, telling the difference between G_s^F and a random function of the same size. The second and third conditions above imply that, for many seeds s, G_s^F is close to a function with small C complexity. Thus, the property will not hold for many such functions (as long as close is within the tolerance, and small within the usefulness of the property). On the other hand, largeness implies that it will hold for many random functions. A gap between these two probabilities implies a distinguishing probability. The size of the distinguisher we obtain will depend on the stretch L and the constructivity of the property.

Convert distinguisher to a predictor. We use the contrapositive of the correctness proof of the PRFG construction to obtain a *predictor* that non-trivially predicts F. Note that non-trivially usually means with advantage at most $1/L$ over random guessing, so the smaller the stretch, the better the predictor will be.

Reverse the amplification. Finally, we apply the converse of the hardness amplification correctness proof to obtain a circuit that computes the original function f with good probability. Note that the agreement of the circuit for f will depend on the strength of the hardness amplifier we can use (which is largely determined by the tolerance) but also on the prediction advantage (largely determined by the stretch, itself determined by the usefulness of the property). Thus, the strongest results will only apply when the tolerance is exponentially close to 1/2 and the usefulness is exponential.

3.4 The case of $\mathsf{AC}^0[2]$

We first consider the case of amplification for $\mathsf{AC}^0[2]$. The case of $\mathsf{AC}^0[q]$ for primes $q > 2$ can be done in a similar way, where we work with $\mathbf{GF}(q)$-valued rather than Boolean functions; we sketch the argument in Section 3.5 below.

Given a boolean function $f\colon \{0,1\}^n \to \{0,1\}$, and a parameter $k = k(n) \in \mathbb{N}$, the amplification $\mathsf{Amp}_k(f)$ is defined as the Goldreich-Levin (Hadamard code) encoding of the k-wise direct product of f:

$$\textsc{Amp}_k(f) = F(x_1, \dots, x_k, b_1, \dots, b_k) = \sum_{i=1}^{k} b_i \cdot f(x_i),$$

where $x_1, \dots, x_k \in \{0,1\}^n$, $b_1, \dots, b_k \in \{0,1\}$, and the summation is modulo 2.

It is shown in [4] that the error parameter of the learning algorithm for f is a function of k and the stretch L of the generator.

▶ **Theorem 6** ([4]). *Suppose the NW generator based on the function $F = \textsc{Amp}_k(f)$, with output strings of length L, is broken with a constant distinguishing probability. Then, using the distinguisher and membership queries to f, one can construct a circuit computing f on at least $1 - \epsilon$ fraction of inputs, for $\epsilon \leq O((\ln L)/k)$. The construction algorithm is a randomized $\mathsf{poly}(n, k, L)$-time algorithm.*

Suppose there is a function $h \in \mathsf{AC}^0[2]$ such that $\operatorname{dist}(f, h) = \beta$. As observed in [4], the function $H = \mathsf{Amp}_k(h) \in \mathsf{AC}^0[2]$ for any $k = k(n) \leq \mathsf{poly}(n)$. It is also easy to argue that $\operatorname{dist}(F, H) = 1/2 - (1-\beta)^k/2$. For a given $\tau = \tau(\ell)$, we want to choose k so that $\operatorname{dist}(F, H) \leq \tau/4$. That is, we want $(1-\beta)^k \geq 1 - \tau/2$. Using the inequalities $1 + x \leq e^x$ (true for all x), and $1 - x \geq e^{-2x}$ (true for all $0 \leq x \leq 0.7$), we are allowed to take $k = \tau/(4\beta)$.

Then the NW generator based on F outputs a truth table of an ℓ-variate function that has the expected (over random seeds z to the generator) relative distance at most $\tau/4$ from the class of $\mathsf{AC}^0[2]$ circuits of size u, for weakly-exponential circuit size u (for which we have a tolerant natural property given by Theorem 5). By Markov's inequality, we get that the actual distance is at most τ for at least 3/4 fraction of the random seeds z to the generator.[2] Thus, for $\mathsf{AC}^0[2]$, we can make the stretch L of our generator to be quasipolynomial, $L = \exp(\mathsf{poly}(\log n))$. Then $\ell = \log L = \mathsf{poly}(\log n)$.

[2] Here, and for the case of $\mathsf{AC}^0[q]$ for primes $q > 2$ later, we can use a simple averaging argument and keep the NW generator as is, because we have natural properties for these classes with very poor tolerance parameters. However, for the general case, when we may have better tolerance parameters, we achieve better concentration by combining the NW generator with a pairwise-independent generator.

As we have $(1/\ell^3)$-tolerant natural property for $\mathsf{AC}^0[2]$ circuits of size u computing ℓ-input boolean functions (Theorem 5), we set $\tau = (1/\ell^3)$, and get that $k = (4\beta\ell^3)^{-1}$. As the τ-tolerant natural property breaks the NW generator based on F, we get by Theorem 6 that f can be learned up to the error $O((\log L)/k) \leq O(\beta \cdot \ell^4) \leq \mathsf{poly}(\log n) \cdot \beta$.

Thus we have proved the following.

▶ **Theorem 7** (Agnostic learning of $\mathsf{AC}^0[2]$)**.** *There is a randomized quasipolynomial-time algorithm for agnostically learning, with membership queries, a function* $f\colon \{0,1\}^n \to \{0,1\}$ *with* $\mathrm{dist}(f, \mathsf{AC}^0[2]) \leq \beta$ *(for a non-negligible* $\beta > 0$*), producing a circuit that computes* f *on all but at most* $\mathsf{poly}(\log n) \cdot \beta$ *fraction of inputs.*

3.5 The case of $\mathsf{AC}^0[q]$ for prime $q > 2$

Next, we consider the case of agnostic learning for $\mathsf{AC}^0[q]$ for prime $q > 2$. While this follows the general outline of the $\mathsf{AC}^0[2]$ case, there are some differences. In particular, to keep the function generators close to functions in $\mathsf{AC}^0[q]$, we need to consider them as producing functions which take Boolean $\{1, -1\}$ inputs to outputs in the range $\{0, \ldots, q-1\}$ of integers modulo q. We need to adjust the natural property from [28] to handle such functions. This turns out to actually simplify the argument from [28] and to eliminate one step (the von Neumann construction) from the PRFG construction in [4].

Our learning algorithm follows the general outline.

Preconditions. We assume membership query access to a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, and a value β and integer d so that we are promised that there is an h in $\mathsf{AC}^0[q]$ computable by a depth d circuit and $\mathrm{dist}(f, h) \leq \beta$.

Amplification. Given a parameter $k = k(n)$, the mod q amplification $\mathsf{Amp}_{k,q}(f)$ is defined as the mod q Goldreich-Levin (Hadamard code) encoding of the k-wise direct product of f:

$$\textsc{Amp}_{k,q}(f) = F(x_1, \ldots, x_k, b_1, \ldots, b_k) = \sum_{i=1}^{k} b_i \cdot f(x_i),$$

where $x_1, \ldots, x_k \in \{1, -1\}^n$, $b_1, \ldots, b_k \in \{0, \ldots, q-1\}$, and the summation is modulo q. Note that this function takes on values in $\{0, \ldots, q-1\}$. We will extend the class $AC^0[q]$ to include such functions in any of several obvious ways, e.g., by having q output gates with the one true one selecting the output. We can code inputs taking on such values similarly.

In our construction, we will set $k = 1/(10 \cdot \beta)$. Let the functions H and F be defined by $H = \mathsf{Amp}_{k,q}(h) \in \mathsf{AC}^0[q]$ and $F = \mathsf{Amp}_{k,q}(f)$. Then $\mathrm{dist}(F, H) = (1-(1-\beta)^k)(1-1/q) \leq k\beta = .01$, since if f and h agree on all k inputs, the functions F and H will agree, and otherwise, they agree with conditional probability $1/q$. Also, H is computable by a depth $d+2$ $\mathsf{AC}^0[q]$ circuit of polynomial size, and a query to F can be simulated with k queries to f.

Pseudo-random function generator. As in [4], we use a version of the NW generator with a design based on polynomials over $\mathbf{GF}(q)$. We are applying this to the function F with non-Boolean outputs from $\mathbf{GF}(q)$, so the resulting truth table will be, for each seed s, a vector of values mod q. We will set the stretch L to be quasi-polynomial in n, $L = \exp(C \cdot \log^{qd+c} n)$ for some constants C and c, where we need the q in the exponent of the $\mathsf{polylog}$ because of the overhead of GL reconstruction for circuits with outputs in $\mathbf{GF}(2)$. Note that we can construct such a truth table with L queries to F. A subtlety is that, while we look at the sets in the design as determined by polynomials over $\mathbf{GF}(q)$, we only consider those L polynomials of degree $\ell - 1$, where $\ell = \log_2 L$, with co-efficients in $\{1, -1\}$.

Call this pseudo-random function generator using F and H respectively, and seed s, G_s^F and G_s^H. As noted in [4], for each seed s, G_s^H can be computed by $\mathsf{poly}(n)$ sized circuits of depth $d + O(1)$.

Since for a random seed s and random position I, the value F is queried at is uniform, $\mathbb{E}_s\left[\mathrm{dist}(G_s^F, G_s^H)\right] = \mathrm{dist}(F, H) \leq .01$. By Markov, we get $\Pr\left[\mathrm{dist}(G_s^F, G_s^H) \geq .1\right] \leq .1$.

Apply natural property. At this point, we apply a tolerant natural property. We need a variant of natural property that applies to functions with Boolean inputs and outputs in $\mathbf{GF}(q)$. It turns out that the Razborov-Rudich [28] natural property for $\mathsf{AC}^0[q]$ is actually simpler in this case. We prove the following in Section B of the appendix.

▶ **Lemma 8** (Tolerant natural property for $\mathsf{AC}^0[q]$)**.** *There is a* P*-natural property* $\{R_n\}_{n\geq 0}$ *with largeness* $1/2$*, and* $(.15)$*-tolerant* $\exp(\Omega(n^{1/(2d)}))$*-usefulness against* d*-depth* $\mathsf{AC}^0[q]$ *circuits computing functions* $f\colon \{1,-1\}^n \to \mathbf{GF}(q)$*.*

We get that at most $1/10$ of the functions G_s^F will be of high complexity, whereas a random function will be of high complexity with probability $1/2$. So testing whether a function has high complexity gives us a $\mathsf{poly}(L)$ size distinguisher with constant advantage for distinguishing G_s^F from a random function.

Converting to a predictor. Using the standard hybrid argument and proof of correctness for the NW generator, we can convert this distinguisher into a predictor circuit of size $\mathsf{poly}(L)$ and advantage $\Omega(1/L)$ of predicting $F(z)$ over random guessing. (To compute this predictor, we need to query F and hence f at $\mathsf{poly}(L)$ positions; see [4]. This is the main step that requires membership queries.)

Converse of amplification. Applying the converse of the generalized GL construction and the direct product theorems, we can convert this predictor circuit into one that computes f on $1-\gamma$ inputs, where $(1-\gamma)^{\Omega k} = \Omega(1/L)$. Thus, $e^{-C_1\gamma k} = C_2/L$, or $\gamma = O(\log L/k) = O(\beta \cdot \log L) = O(\beta \cdot \log^{qd+c} n)$. So we get an agnostic learner that works in time and queries quasi-polynomial in n, and with error at most $O(\log^{qd+c} n) \cdot \beta$. (Note that this assumes β is non-negligible; otherwise, the time and circuit size depend on $1/\beta$ as well).

Combining all these pieces, we have the following.

▶ **Theorem 9** (Agnostic learning of $\mathsf{AC}^0[q]$)**.** *Let $q > 2$ be any prime. There is a randomized quasipolynomial-time algorithm for agnostically learning, with membership queries, a function* $f\colon \{0,1\}^n \to \{0,1\}$ *with* $\mathrm{dist}(f, \mathsf{AC}^0[q]) \leq \beta$ *(for a non-negligible* $\beta > 0$*), producing a circuit that computes* f *on all but at most* $\mathsf{poly}(\log n) \cdot \beta$ *fraction of inputs.*

4 Agnostic learning from tolerant natural properties

Next, we consider the case of agnostic learning for any Λ closed under $\mathsf{AC}^0[2]$-reductions for *any* natural property against Λ with super-constant tolerance and usefulness. This follows the general outline of the $\mathsf{AC}^0[2]$ case, but we need to use a variant of the NW pseudorandom generator to take advantage of (potentially) better tolerance. We will use Chebyshev instead of Markov to bound the probability, over random seeds z, that the functions mapped to by the generator have small distance. Our generic learning algorithm also follows the outline.

Preconditions. Let Λ be some complexity class closed under $\mathsf{AC}^0[2]$-reductions. Let $\mathcal{R}$ be a BPP-constructive, τ-tolerant, u-useful natural property against Λ, for super-constant

τ with largeness $\delta > (1/2)$. Write $\tau = (1/2) - \tau'$, because it will sometimes be easier to work with τ as an "advantage." We assume membership query access to a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, and a value β so that we are promised that there is an h in Λ with $\mathrm{dist}(f, h) \leq \beta$.

Amplification. We use $\textsc{Amp}_k$ identically to the specific case of $\mathsf{AC}^0[2]$, except that we set k later based on abstract τ and u. Let $F = \textsc{Amp}_k(f)$, $H = \textsc{Amp}_k(h)$, as before $\mathrm{dist}(F, H) = (1/2) - (1/2)(1 - \beta)^k$, which we call μ.

Pseudo-random function generator. As in [4], we use a version of the NW generator with a design based on polynomials over $\mathbf{GF}(2)$. Recall that the NW design for parameters $n, m, L \in \mathbb{N}$ is a family of sets $S_1, \ldots, S_L \subseteq [m]$, of size $|S_i| = n$, for all $1 \leq i \leq L$, and small overlap $|S_i \cap S_j| \leq \log L = \ell$ for all $1 \leq i \neq j \leq L$. It was shown in [4] that such designs can be efficiently locally computed by $\mathsf{AC}^0[q]$ circuits, for any prime q.

▶ **Lemma 10** (NW design in $\mathsf{AC}^0[q]$ [25, 4]). *Let q be any prime. There is a constant $d_0 \in \mathbb{N}$ such that, for any n and $L < 2^n$, there exists an NW design $S_1, \ldots, S_L$ with parameters as defined above, so that the function $MX_{NW}\colon \{0,1\}^\ell \times \{0,1\}^m \to \{0,1\}^n$, defined by $MX_{NW}(i, z) = z|_{S_i}$, where $z|_{S_i}$ denotes the substring of z indexed by S_i, is computable by an $\mathsf{AC}^0[q]$ circuit of depth d_0 and size $\mathsf{poly}(\ell, n)$.*

The NW generator [25] based on a boolean function $F\colon \{0,1\}^n \to \{0,1\}$ is $G^F\colon \{0,1\}^m \to \{0,1\}^L$ defined as $G^F(z) = F(z|_{S_1}) \circ \cdots \circ F(z|_{S_L})$, where $S_1, \ldots, S_L$ is the NW design as above. Lemma 10 implies that if $F \in \mathsf{AC}^0[2]$, then, for each seed z, the output $G^F(z)$ is the truth table of an ($\ell = \log L$)-variate Boolean function of $\mathsf{AC}^0[2]$ circuit complexity at most $\mathsf{poly}(\ell, n)$.

Let $H\colon \{0,1\}^n \to \{0,1\}$ be another boolean function such that $\mathrm{dist}(F, H) \leq \mu$, for some $\mu \in [0,1]$. By the definition of the NW generator, we have that the expected hamming distance between the L-bit strings $G^F(z)$ and $G^H(z)$, over random seeds z, is $\mathrm{dist}(F, H) \cdot L \leq \mu \cdot L$. For our agnostic learning framework, it is important (as explained in the previous section) that the NW generator have the *concentration property*: for most seeds z, the hamming distance between $G^F(z)$ and $G^H(z)$ is close to the expected distance $\mu \cdot L$.

We achieve this concentration property by adding a pairwise-independent string generator as a component of the NW generator. Let $PI\colon \{0,1\}^\ell \times \{0,1\}^{m'} \to \{0,1\}^n$ be a *pairwise independent generator* such that

1. for each $i \in [L]$, the distribution $PI(i, z)$ over uniformly random $z \in \{0,1\}^{m'}$ is uniform over $\{0,1\}^n$, and
2. for all $i \neq j \in [L]$, the distribution of $PI(i, z)$ and $PI(j, z)$, over uniformly random seeds $z \in \{0,1\}^{m'}$, is uniform over $\{0,1\}^n \times \{0,1\}^n$.

Such generators exist for $m' \leq n(\ell + 1)$; for example, pick a random 0/1 matrix A of dimension $n \times \ell$ and a random 0/1 vector v of dimension n. Let $z = (A, v)$. Define $P(i, (A, v)) = A \cdot i + v$, where $A \cdot i$ denotes the matrix-vector multiplication, and all operations are over $\mathbf{GF}(2)$. It is easy to see that this generator $PI(i, z)$ is computable by an $\mathsf{AC}^0[2]$ circuit of polynomial size.

Define the modified NW generator $G'^F\colon \{0,1\}^m \times \{0,1\}^{m'} \to \{0,1\}^L$, based on the n-variate boolean function F, as follows:

$$G'^F(z_1, z_2) = F(z_1|_{S_1} \oplus PI(1, z_2)) \circ \cdots \circ F(z_1|_{S_L} \oplus PI(L, z_2)),$$

where S_i's form the NW design, and PI is the pairwise-independent generator as above, and $\oplus$ denotes the bit-wise XOR of the corresponding n-bit strings.

Observe that since the generator PI is efficiently locally computable in $\mathsf{AC}^0[2]$, we still get (by Lemma 10) that the ℓ-bit function output by G'^F, for $F \in \mathsf{AC}^0[2]$, has $\mathsf{AC}^0[2]$ circuit complexity at most $\mathsf{poly}(\ell, n)$. Next, the generator G' allows the same kind of reconstruction as the original NW generator: given a distinguisher for G' with a constant distinguishing probability, one can efficiently construct (using membership queries to F) a small circuit computing F on at least $1/2 + \Omega(1/L)$ fraction of inputs. Finally, the generator $G'^F(z_1, z_2)$, for uniformly random seeds z_1 and z_2, outputs L values of F on *pairwise-independent* uniformly random n-bit inputs.

From pairwise independence we get that the hamming distance between $G'^F(z_1, z_2)$ and $G'^H(z_1, z_2)$, over random z_1 and z_2, is concentrated around the expectation, by the Chebyshev bound. More precisely, for F and H with $\text{dist}(F, H) \leq \mu$, we have by Chebyshev that

$$\mathbf{Pr}_z \left[\left| \text{DIST}(G'^F(z), G'^H(z)) - \mu \cdot L \right| > \zeta \cdot L \right] < \frac{1}{\zeta^2 \cdot L},$$

which we will require to be less than $1/4$. We parameterize the bound with $\zeta = (1/4)(1-\beta)^k$. For the selected ζ, and the stretch L we are forced to pick later, this is immediate.

Apply natural property. At this point, we apply a tolerant natural property to produce a distinguisher circuit for the generator above. This induces the following system of constraints, which relate the usefulness, tolerance, and density of the property to the stretch and concentration of the generator. Let $\Lambda\text{-SIZE}(G'^H(z)) = s_H$. We require that $s_H \leq u(\ell)$, to respect the size lower bound. We re-arrange the Chebyshev bound above and see that we should require $\mu + \zeta < \tau(\ell)$, respect tolerance, and ensure a good distinguishing gap from the property. We satisfy the first requirement by setting $\ell \geq u^{-1}(s_H)$. The second one is equivalent to $(1/4)(1-\beta)^k > \tau'(\ell)$. In this case, the tolerant property can only accept $G^F(z)$ with probability $(1/4)$ but accepts a random function with probability at least $(1/2)$, giving us a $(1/4)$ distinguishing gap. We can satisfy both constraints by setting $k = \Theta(\log(\tau'(\ell))/\beta)$.

Converting to a predictor. Using a small modification of the standard hybrid argument and proof of correctness for the NW generator, we can convert this distinguisher into a predictor circuit of size $\mathsf{poly}(L)$ and advantage $\Omega(1/L)$ of predicting $F(z)$ over random guessing. The modified predictor just embeds a construction of PI and shifts/unshifts inputs to the distinguisher circuit as necessary. (To compute this predictor, we need to query F and hence f at $\mathsf{poly}(L)$ positions; see [4]. This is the main step that requires membership queries.) From this step we know that our runtime is at most $\mathsf{poly}(L)$, and the circuit output at this stage is already size $\mathsf{poly}(L)$.

Converse of amplification. Identical to the case of $\mathsf{AC}^0[q]$, but with the additional constraints mentioned above. Note that the runtime of these algorithms is randomized time in the size of the input circuit, so runtime, number of queries, and output circuit size of this stage will also be dominated by L. Use of this algorithm imposes the following constraint from the direct product reconstruction stage: $\mathsf{poly}(1/L) > e^{-k\epsilon/c}$. So $\epsilon > \Theta(\log(L)/k)$. Substituting in our value for k, this gives us $\epsilon = \Theta(\ell\beta/\log(\tau'(\ell)))$ for $\ell = u^{-1}(s_H)$.

Summarizing, we get a generic reduction from tolerant natural properties to agnostic learning.

▶ **Theorem 11** (Tolerant natural properties imply agnostic learning algorithms)**.** *Let $\mathcal{R}$ be a natural property against Λ closed under $\mathsf{AC}^0[2]$ reductions with $(1/2-\tau')$-tolerant u-usefulness and* *largeness $\delta \geq 1/2$. Then there is a randomized algorithm such that, for any n-ary boolean functions f and h with* $\mathrm{dist}(f, h) < \beta$ *and $s_h = \Lambda$-SIZE(h), the algorithm, given oracle access to f, produces a circuit ϵ-approximating f, for any $\epsilon > \beta \cdot u^{-1}(\mathsf{poly}(s_h))/\log(\tau'(u^{-1}(\mathsf{poly}(s_h))))$, in time* $\mathsf{poly}(\max\{2^{u^{-1}(\mathsf{poly}(s_h))},\ 1/\epsilon\})$.

In particular, this means if we have a “perfect” natural property, with exponential usefulness u and inverse exponential tolerance τ', we have a polynomial-time learning algorithm with error bound $\Theta(\beta)$. Thus Theorem 2 is a special case of Theorem 11.

5 Hardness of removing membership queries

Is it possible to eliminate membership queries from our algorithm, learning just from random examples? We note that removing membership queries would give us quasipolynomial-time algorithms for two notoriously difficult problems: learning parities with noise (LPN) for the case of $\mathsf{AC}^0[2]$ and a variant of learning with errors (LWE) for $\mathsf{AC}^0[q]$.

Though learning parities with noise under uniform distribution can be done in polynomial time with membership queries (by the Goldreich-Levin algorithm [11]), without membership queries this problem is believed to be hard. Learning parities with noise efficiently under uniform distribution would give learning algorithms for DNFs and k-juntas (and in general, for any problem reducible to finding a heavy Fourier coefficient of a function) [8].

In the worst case, LPN is known to be NP-hard (and MAX-SNP-hard). The average-case hardness of LPN has been considered as early as 1993, when Blum, Furst, Kearns and Lipton have given a simple construction of a pseudorandom bit generator based on the assumption that learning parities with constant noise rate is hard [2]. In practical cryptography, average-case hardness of LPN is the basis for Hopper and Blum authentication protocol [14]. There, the noise rate is usually set to a constant $\eta \in (0, 1/2)$, in particular $\eta = 1/8$ has been used in applications [22]. Though for $\mathsf{AC}^0[2]$ our algorithm works for noise up to $1/\mathsf{polylog}(n)$, we can tolerate constant noise for $\mathsf{AC}^0[q]$.

Hardness of LWE problem follows from worst-case hardness of variants of the lattice shortest vector problem [29]. Whereas LPN has been used to build "minicrypt" cryptographic primitives, LWE has been used for public-key cryptosystems [1, 29].

6 Open questions

While there are correlation bounds for $\mathsf{AC}^0[q]$ circuits that say that some explicit functions cannot be computed by “small” circuits on significantly more that $1/2 + 1/\sqrt{n}$ fraction of inputs, we do not know how to get natural properties with tolerance close to 1/2. Getting natural properties with better tolerance parameters would immediately imply improved parameters for our agnostic learning algorithms for the corresponding circuit classes. (Of course, getting stronger correlation bounds for $\mathsf{AC}^0[q]$, whether obtained by natural proofs or not, is in itself a very important problem in circuit complexity.)

Can one get a query agnostic learning algorithm for $\mathsf{AC}^0[q]$ with the optimal error $\mathsf{opt} + \epsilon$? It seems that, even with ideal tolerance and usefulness, our approach of getting learning algorithms from natural properties will at best achieve the error $O(\mathsf{opt}) + \epsilon$. So one needs a new approach, perhaps inspired by the learning algorithm in this paper.

In fact, probably the main open problem is to get a more “natural” (understandable) learning algorithm for $\mathsf{AC}^0[q]$ than our construction, which combines the NW-style generator analysis with circuit lower bound proofs. As a possible first step, it would be interesting to get an alternative agnostic learning algorithm for low-degree polynomials over $\mathbf{GF}(2)$.

References

1 Michael Alekhnovich. More on average case vs approximation complexity. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 298–307. IEEE, 2003.

2 Avrim Blum, Merrick Furst, Michael Kearns, and Richard J Lipton. Cryptographic primitives based on hard learning problems. In *Annual International Cryptology Conference*, pages 278–291. Springer, 1993.

3 Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

4 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In Ran Raz, editor, *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, volume 50, pages 10:1–10:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016.

5 Ruiwen Chen, Valentine Kabanets, Antonina Kolokolova, Ronen Shaltiel, and David Zuckerman. Mining circuit lower bound proofs for meta-algorithms. *Computational Complexity*, 24(2):333–392, 2015.

6 Vitaly Feldman. On the power of membership queries in agnostic learning. *Journal of Machine Learning Research*, 10:163–182, 2009.

7 Vitaly Feldman. Distribution-specific agnostic boosting. In *Innovations in Computer Science – ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 241–250, 2010.

8 Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. New results for learning noisy parities and halfspaces. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 563–574. IEEE, 2006.

9 Jean-Bernard Fischer and Jacques Stern. An efficient pseudo-random generator provably as secure as syndrome decoding. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 245–255. Springer, 1996.

10 Sally A. Goldman, Michael J. Kearns, and Robert E. Schapire. On the sample complexity of weakly learning. *Inf. Comput.*, 117(2):276–287, 1995.

11 Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 25–32. ACM, 1989.

12 Parikshit Gopalan, Adam Tauman Kalai, and Adam R. Klivans. Agnostically learning decision trees. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 527–536. ACM, 2008.

13 Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001.

14 Nicholas J. Hopper and Manuel Blum. Secure human identification protocols. In *Advances in Cryptology – ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, pages 52–66, 2001.

15 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 961–972. SIAM, 2012.

16 Russell Impagliazzo, Raghu Meka, and David Zuckerman. Pseudorandomness from shrinkage. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 111–119. IEEE Computer Society, 2012.

17 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In Frank Thomson Leighton and Peter W. Shor, editors, *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997*, pages 220–229. ACM, 1997.

18 Jeffrey C. Jackson. Uniform-distribution learnability of noisy linear threshold functions with restricted focus of attention. In *Proceedings of the 19th Annual Conference on Learning Theory*, COLT'06, pages 304–318, Berlin, Heidelberg, 2006. Springer-Verlag.

19 Adam Tauman Kalai, Adam R. Klivans, Yishay Mansour, and Rocco A. Servedio. Agnostically learning halfspaces. *SIAM J. Comput.*, 37(6):1777–1805, 2008.

20 Michael J. Kearns, Robert E. Schapire, and Linda Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

21 Wee Sun Lee, Peter L. Bartlett, and Robert C. Williamson. Efficient agnostic learning of neural networks with bounded fan-in. *IEEE Trans. Information Theory*, 42(6):2118–2132, 1996.

22 Éric Levieil and Pierre-Alain Fouque. An improved lpn algorithm. In *International Conference on Security and Cryptography for Networks*, pages 348–359. Springer, 2006.

23 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993.

24 Vadim Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 378–389. Springer, 2005.

25 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

26 Krzysztof Pietrzak. Cryptography from learning parity with noise. In *International Conference on Current Trends in Theory and Practice of Computer Science*, pages 99–114. Springer, 2012.

27 A. A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

28 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.

29 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34, 2009.

30 Rahul Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 183–192. IEEE Computer Society, 2010.

31 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 77–82, 1987.

32 Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, November 1984.

33 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 231–240. ACM, 2010.

34 Ryan Williams. Non-uniform ACC circuit lower bounds. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011, San Jose, California, June 8-10, 2011*, pages 115–125. IEEE Computer Society, 2011.

A Tolerant natural property for $\mathsf{AC}^0[2]$

Razborov [27] showed the following natural property for $\mathsf{AC}^0[2]$:

> Given an n-variate boolean function f, construct certain matrices $A_1, \ldots, A_b$, for $b = n/2 - \sqrt{n}$, of dimensions at most $2^n \times 2^n$, and check if at least one of the matrices has rank at least $2^n/(140 \cdot n^2)$ over $\mathbf{GF}(2)$.

More precisely, for $a = n/2 - \sqrt{n}$ and all $i \le a$, define A_i to be the matrix whose rows are labeled by size a subsets of $[n]$, and whose columns are labeled by size i subsets of $[n]$. For $K \subseteq [n]$, let $Z(K) = \{x \in \{0,1\}^n \mid x|_K = \vec{0}\}$. For a row $I \subseteq [n]$ and a column $J \subseteq [n]$, define $(A_i)_{I,J} = \oplus_{x \in Z(I \cup J)} f(x)$.

It is possible to show that at least 1/2 of all n-variate boolean functions satisfy this property; so we have largeness (see [4]). The usefulness of this property is due to the following two lemmas. Below we denote by $\mathcal{P}(D)$ the linear space of all n-variate degree D multilinear polynomials over $\mathbf{GF}(2)$.

▶ **Lemma 12** ([27]). *For an n-variate boolean function f and the corresponding matrices $A_1, \ldots, A_b$, for $b = n/2 - \sqrt{n}$, we have for all $1 \le i \le b$ that*

$$\mathrm{DIST}(f, \mathcal{P}(\sqrt{n})) \ge \mathrm{rank}(A_i).$$

▶ **Lemma 13** ([27]). *For an n-variate boolean function f, if f is computable by a d-depth $\mathsf{AC}^0[2]$ circuit of size s, then*

$$\mathrm{dist}(f, \mathcal{P}((O(\log(s/\epsilon))^d)) \le \epsilon.$$

So for $\epsilon = 1/n^3$ and size $s < \exp(\Omega(n^{1/(2d)}))/n^3$, we get by Lemma 13 that any f computable by a d-depth $\mathsf{AC}^0[2]$ circuit of size s is such that $\mathrm{DIST}(f, \mathcal{P}(\sqrt{n})) \le 2^n/n^3$. Hence, by Lemma 12, all the corresponding matrices A_i for f have rank at most $2^n/n^3 \le 2^n/(140 \cdot n^2)$ (for all sufficiently large n), and so f is rejected by the natural property.

Now suppose that h is an n-variate boolean function that is close to f, i.e., for some $0 \le \beta \le 1$,

$$\mathrm{dist}(h, f) \le \beta,$$

where f is as above. Then we get by the triangle inequality that

$$\mathrm{DIST}(h, \mathcal{P}(\sqrt{n})) \le (\beta + n^{-3}) \cdot 2^n,$$

which, in particular, means that for any $\beta \le 1/n^3$, such a function h will also be rejected by the natural property above.

Thus we have proved the following.

▶ **Lemma 14** (Tolerant natural property for $\mathsf{AC}^0[2]$). *There is a P-natural property $\{R_n\}_{n \ge 0}$ with largeness 1/2, and $(1/n^3)$-tolerant $\exp(\Omega(n^{1/(2d)}))$-usefulness against d-depth $\mathsf{AC}^0[2]$ circuits.*

B Tolerant natural property for $\mathsf{AC}^0[q]$ for prime $q > 2$

Here we prove the following.

▶ **Lemma 15** (Tolerant natural property for $\mathsf{AC}^0[q]$). *There is a* P*-natural property* $\{R_n\}_{n\geq 0}$ *with largeness* $1/2$*, and* $(.15)$*-tolerant* $\exp(\Omega(n^{1/(2d)}))$*-usefulness against* d*-depth* $\mathsf{AC}^0[q]$ *circuits computing functions* $f\colon \{1,-1\}^n \to \mathbf{GF}(q)$.

Proof. Let $\mathcal{M}$ be the vector space of all n-variate multilinear polynomials over $GF(q)$, and let $\mathcal{L}$ be the subspace of those polynomials of degree at most $n/2$. Given such a multilinear polynomial f (and any truth table indexed by $\{1,-1\}^n$ over $GF(q)$ defines such a polynomial), we say that f is high complexity if the dimension $\dim(\mathcal{L} + f \cdot \mathcal{L}) \geq 3/4 \cdot N$, where $N = 2^n$.

Note that, for any function f of degree d, $\mathcal{L} + f \cdot \mathcal{L}$ is contained within the space of multilinear polynomials of degree $l/2 + d$, which has dimension at most $N(1/2 + O(d/\sqrt{n}))$. Changing any D values can increase this dimension by at most D (since adding the dimension D vector space of all functions on these D points to the subspace for the original function includes the subspace functions for the changed function). So in particular, any high complexity function must have distance at least $1/5$ from any function of degree $c\sqrt{n}$ for some $c > 0$. Since by work by Razborov [27] and Smolensky [31], any function in $\mathsf{AC}^0[q]$ of depth d and size s is within ϵ distance of a multilinear polynomial over $\mathbf{GF}(q)$ of degree $O(\log(s/\epsilon)^d)$, any high complexity function must be distance .15 from any function computed by size $\exp(\Omega(n^{1/(2d+C)}))$ depth $d + C$ circuits with mod q gates.

At least half of such functions have high complexity. From [31], for p the product of all l inputs (i.e., the parity of the number of -1 inputs), $\mathcal{L} + p \cdot \mathcal{L} = \mathcal{M}$. Then for f any function, either f has high complexity or $p - f$ does. Because if both have low complexity, then

$$\dim(\mathcal{L} + f \cdot \mathcal{L}) = \dim \mathcal{L} + \dim((f \cdot \mathcal{L})/\mathcal{L}) < \frac{3}{4} \cdot N,$$

so $\dim((f \cdot \mathcal{L})/\mathcal{L}) < (1/4) \cdot N$, and similarly for $p - f$. Then

$$\begin{aligned}
\dim \mathcal{M} &= \dim(\mathcal{L} + p \cdot \mathcal{L}) \\
&\leq \dim(\mathcal{L} + f \cdot \mathcal{L} + (p - f) \cdot \mathcal{L}) \\
&\leq \dim \mathcal{L} + \dim((f \cdot \mathcal{L})/\mathcal{L}) + \dim(((p - f) \cdot \mathcal{L})/\mathcal{L}) \\
&< N/2 + N/4 + N/4 \\
&= N,
\end{aligned}$$

a contradiction. Since all functions can be paired up into $f, p - f$ pairs, at least half the functions have high complexity. Clearly, we can test whether a function has high complexity in $\mathsf{poly}(N)$ time. ◀

On the Complexity of Constrained Determinantal Point Processes

L. Elisa Celis[1], Amit Deshpande[2], Tarun Kathuria[3], Damian Straszak[4], and Nisheeth K. Vishnoi[5]

1 **École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland**
2 **Microsoft Research, Bangalore, India**
3 **Microsoft Research, Bangalore, India**
4 **École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland**
5 **École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland**

Abstract

Determinantal Point Processes (DPPs) are probabilistic models that arise in quantum physics and random matrix theory and have recently found numerous applications in theoretical computer science and machine learning. DPPs define probability distributions over subsets of a given ground set, they exhibit interesting properties such as negative correlation, and, unlike other models of negative correlation such as Markov random fields, have efficient algorithms for sampling. When applied to kernel methods in machine learning, DPPs favor subsets of the given data with more diverse features. However, many real-world applications require efficient algorithms to sample from DPPs with additional constraints on the sampled subset, e.g., partition or matroid constraints that are important from the viewpoint of ensuring priors, resource or fairness constraints on the sampled subset. Whether one can efficiently sample from DPPs in such constrained settings is an important problem that was first raised in a survey of DPPs for machine learning by Kulesza and Taskar and studied in some recent works.

The main contribution of this paper is the first resolution of the complexity of sampling from DPPs with constraints. On the one hand, we give exact efficient algorithms for sampling from constrained DPPs when the description of the constraints is in unary; this includes special cases of practical importance such as a small number of partition, knapsack or budget constraints. On the other hand, we prove that when the constraints are specified in binary, this problem is **#P**-hard via a reduction from the problem of computing mixed discriminants; implying that it may be unlikely that there is an FPRAS. Technically, our algorithmic result benefits from viewing the constrained sampling problem via the lens of polynomials and we obtain our complexity results by providing an equivalence between computing mixed discriminants and sampling from partition constrained DPPs. As a consequence, we obtain a few corollaries of independent interest: 1) An algorithm to count, sample (and, hence, optimize) over the base polytope of regular matroids when there are additional (succinct) budget constraints and, 2) An algorithm to evaluate and compute mixed characteristic polynomials, that played a central role in the resolution of the Kadison-Singer problem, for certain special cases.

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems, G.2.1 Counting Problems

Keywords and phrases determinantal point processes, constraints, matroids, sampling and counting, polynomials, mixed discriminant

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.36

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 36; pp. 36:1–36:22
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Algorithms for sampling from a discrete set of objects are sought after in various disciplines of computer science, optimization, mathematics and physics due to their far reaching applications. For instance, sampling from the Gibbs distribution was one of the original optimization methods (see, e.g., [17]) and sampling from dependent distributions is often used in the design of approximation algorithms (see, e.g., [5, 8, 19]). In machine learning, algorithms for sampling from discrete probability distributions are desired in various summarization, inference and learning tasks [33, 28, 24]. A particular class of probability distributions that has received much attention are the Determinantal Point Processes (DPP). In the discrete setting, a DPP is a distribution over subsets of a finite data set $[m] \overset{\text{def}}{=} \{1, 2, \ldots, m\}$. Here, a data point i is associated to a feature vector $v_i \in \mathbb{R}^d$, and an $m \times m$ positive semidefinite (PSD) kernel L gives the dot product of the feature vectors of any two data points as a measure of their pairwise similarity. Determinants, then, provide a natural measure of the diversity of a subset of data points, often backed by a physical intuition based on volume or entropy. A DPP is thus defined with respect to the kernel L such that for all $S \subseteq [m]$ we have $\mathbb{P}(S) \propto \det(L_{S,S})$, where $L_{S,S}$ is the principal minor of L corresponding to rows and columns from S.[1] The quantity $\det(L_{S,S})$ can be interpreted as the squared volume of the $|S|$-dimensional parallelepiped spanned by the vectors $\{v_i : i \in S\}$ and, intuitively, the larger the volume, the more *diverse* the set of vectors. Hence such distributions tend to prefer most diverse or *informative* subsets of data points. Mathematically, the fact that the probabilities are derived from determinants allows one to deduce elegant and non-trivial properties of such distributions, such as negative correlation and concentration of measure. Efficient polynomial time algorithms for sampling from DPPs (see [20, 10]) is what sets them apart from the other probabilistic models of negative correlation such as Markov random fields. As a consequence, sampling from DPPs has been successfully applied to a number of problems, such as document summarization, sensor placement and recommendation systems [26, 23, 37, 36, 35].

Given the wide applicability of DPPs, a natural question is whether they can be generalized to incorporate priors, budget or fairness constraints, or other natural combinatorial constraints. In other words, given an $m \times m$ kernel L and a family $\mathcal{C} \subseteq 2^{[m]}$ that represents constraints on the subsets, can we efficiently sample from the DPP distribution supported only on $\mathcal{C}$; that is, $\mathbb{P}(S) \propto \det(L_{S,S})$ for $S \in \mathcal{C}$, and $\mathbb{P}(S) = 0$ otherwise. Here are two important special cases.

- *Fairness (or Partition) constraints:* Consider the setting where $[m]$ is a collection of data points and each point is associated with a sensitive attribute such as gender. Then $\mathcal{C}$ is the family of attribute-unbiased subsets of $[m]$ – e.g., those subsets that contain an equal number of male and female points. Thus, the corresponding $\mathcal{C}$-constrained DPP outputs a diverse set of points while maintaining fairness with respect to the sensitive attribute; see [7] for this and other applications of constrained DPPs to eliminating algorithmic bias.
- *Budget constraints:* In data subset selection or active learning, when there is a cost $c_i \in \mathbb{Z}$ associated with each data point, it is natural to ask for a diverse training sample S from a corresponding DPP such that its cost $\sum_{i \in S} c_i$ is bounded from above by $C \in \mathbb{Z}$. See also [34] for a related optimization variant.

[1] We treat DPPs via *L-ensembles*, while commonly they are defined using *kernel matrices*, for practical purposes these two definitions are equivalent.

In their survey, [24] posed the open question of efficiently sampling from DPPs with additional combinatorial constraints on the support of the distribution. Sampling from constrained DPPs is algorithmically non-trivial, as many natural heuristics fail. The probability mass on the constrained family of subsets can be arbitrarily small, hence, ruling out a rejection sampling approach. For partition constraints, a natural heuristic is to sample from independent smaller-sized DPPs, each defined over a different part. However, such a product distribution would select two (potentially very similar) items from two different parts independently, whereas in a constrained DPP distribution they must be negatively correlated. Unlike DPPs and the special case of cardinality-constrained k-DPPs (in which $\mathcal{C}$ is the family of *all* subsets of size k – see Section 1.2), it is not clear that there is a clean expression for the partition function or the marginals of a constrained DPP. Another approach to approximately sample from constrained DPPs is via Markov Chain Monte Carlo (MCMC) methods as in the recent work of [25]. This approach can be shown to be efficient when the underlying Markov chain is connected and the DPP kernel is close to a diagonal matrix (or nearly-log-linear; see Theorem 4 of [25]). However, the above conditions do not hold for sampling partition-constrained subsets – even with constant number of parts – from most DPP kernels. Thus, while the problem of sampling from constrained DPPs has attracted attention, its complexity has remained open.

The main contribution of our paper is the first resolution of the problem of sampling from constrained DPPs. Our results give a dichotomy for the complexity of this problem: On the one hand, we give exact algorithms which are polynomial time when the description of $\mathcal{C}$ (in terms of the costs and budgets) is in *unary*; this includes special cases of practical importance such as the fairness, partition or budget constraints mentioned above. On the other hand, we prove that in general this problem is #**P**-hard when the constraints of $\mathcal{C}$ are specified in binary. Our algorithmic results go beyond the MCMC methods and include special cases of practical importance such as (constantly-many) partition or fairness constraints (studied, e.g., by [7]) and a more general class of budget constraints and linear families defined in the following section.

Our algorithmic results benefit from viewing the probabilities arising in constrained DPPs as coefficients of certain multivariate polynomials. This viewpoint also allows us to extend our result on constrained DPPs to derive important consequences of independent interest. For instance, using the intimate connection between linear matroids and DPPs, we arrive at efficient algorithms to sample a basis of regular matroids when there are additional budget constraints – significantly extending results of [12, 6] for spanning trees. To prove the hardness result, we present an *equivalence* between the problem of *computing* the mixed discriminant of a tuple of PSD matrices and that of *sampling* from partition-constrained DPPs. Mixed discriminants (see Section 4.1 for a definition) generalize the permanent, arise in the proof of the Kadison-Singer problem ([27], see [18] for a survey on this topic) and are closely related to mixed volumes (see, e.g., [4]). However, unlike the result for permanent [21] and volume computation [11], there is evidence that the mixed discriminant problem may be much harder and may not admit an FPRAS; see [15]. Thus, in light of our equivalence between mixed discriminants and partition DPPs, it may be unlikely that we can even approximately sample from partition DPPs (with an arbitrary number of parts) efficiently. Further, this connection implies that important special cases of the mixed discriminant problem, for instance computing the higher order coefficients of the mixed-characteristic polynomial or evaluating the mixed characteristic polynomial of low rank matrices at a given point, can be solved efficiently, which may be of independent interest.

1.1 Our Framework and Results

The starting point of our work is the observation that if we let μ be the measure on subsets of $[m]$ corresponding to the kernel matrix L (i.e., $\mu(S) \stackrel{\text{def}}{=} \det(L_{S,S})$), then given L, there is an efficient algorithm to evaluate the polynomial

$$g_\mu(x) \stackrel{\text{def}}{=} \sum_{S \subseteq [m]} \mu(S) x^S$$

where x^S denotes $\prod_{i \in S} x_i$ for any setting of its variables. Indeed, consider the Cholesky decomposition of the kernel $L = VV^\top$. Then, the polynomial $x \mapsto \det(V^\top X V + I)$ (where X denotes the diagonal matrix with x on the diagonal) is equal to $g_\mu(x)$ (see Fact 8) and hence can be efficiently evaluated using Gaussian elimination for any input x. We say that such a μ has an *efficient evaluation oracle* and, as it turns out, this is the *only* property we need from DPPs and our results generalize to any measure μ for which we have such an evaluation oracle. Before we explain our results, we formally introduce the sampling problem in this general framework.

▶ **Definition 1** (Sampling). Let $\mu : 2^{[m]} \to \mathbb{R}_{\geq 0}$ be a function assigning non-negative real values to subsets of $[m]$ and let $\mathcal{C} \subseteq 2^{[m]}$ be any family of subsets of $[m]$. We denote the (sampling) problem of selecting a set $S \in \mathcal{C}$ with probability $p_S = \frac{\mu(S)}{\sum_{T \in \mathcal{C}} \mu(T)}$ by $\textsc{Sample}[\mu, \mathcal{C}]$.

Building up on the equivalence between sampling and counting [22], we show that if one is given oracle access to the generating polynomial g_μ and if μ is a nonnegative measure, the problem $\textsc{Sample}[\mu, \mathcal{C}]$ is essentially equivalent to the following counting problem; see Theorem 21 in Appendix B.

▶ **Definition 2** (Counting). Let $\mu : 2^{[m]} \to \mathbb{R}_{\geq 0}$ be a function assigning non-negative real values to subsets of $[m]$ and let $\mathcal{C} \subseteq 2^{[m]}$ be any family of subsets of $[m]$. We denote the (counting) problem of computing the sum $\sum_{S \in \mathcal{C}} \mu(S)$ by $\textsc{Count}[\mu, \mathcal{C}]$.

In particular, a polynomial time algorithm for $\textsc{Count}[\mu, \mathcal{C}]$ can be translated into a polynomial time algorithm for $\textsc{Sample}[\mu, \mathcal{C}]$. Interestingly, this relation holds no matter what $\mathcal{C}$ is; in particular, no specific assumptions on how the access to $\mathcal{C}$ is provided are required.

Towards developing counting algorithms in our framework, we focus on a class of families $\mathcal{C} \subseteq 2^{[m]}$, which we call *Budget Constrained Families*, where a cost vector $c \in \mathbb{Z}^m$ and a budget value $C \in \mathbb{Z}$ are given, and the family consists of all sets $S \subseteq [m]$ of total cost $c(S) \stackrel{\text{def}}{=} \sum_{i \in S} c_i$ at most C. We call the counting and sampling problems for this special case $\text{B}\textsc{Count}[\mu, c, C]$ and $\text{B}\textsc{Sample}[\mu, c, C]$ respectively.

Our key result is that the $\text{B}\textsc{Count}$ problem (and hence also $\text{B}\textsc{Sample}$) is efficiently solvable whenever the costs are not too large in magnitude.

▶ **Theorem 3** (Counting under Budget Constraints). *There is an algorithm, which given a function $\mu : 2^{[m]} \to \mathbb{R}$ (via oracle access to g_μ), a cost vector $c \in \mathbb{Z}^m$ and a cost value $C \in \mathbb{Z}$ solves the* $\text{B}\textsc{Count}[\mu, c, C]$ *problem in polynomial time with respect to m and $\|c\|_1$.*

The proof of Theorem 3 (see Section 2) benefits from an interplay between probability measures and polynomials. It reduces the counting problem to computing the coefficients of a certain univariate polynomial which, in turn, can be evaluated efficiently given access to the generating polynomial for μ. We can then employ interpolation in order to recover the required coefficients.

It is not hard to see that Theorem 3 also implies the same result for families with a single *equality* constraint ($c(S) = C$) or for any constraint of the form $c(S) \in K$, where $K \subseteq \mathbb{Z}$ is given as input together with $c \in \mathbb{Z}^m$ and $C \in \mathbb{Z}$. Furthermore, our framework can be easily extended to the case of multiple (constant number of) such constraints.

As mentioned earlier, what makes DPPs attractive is that their generating polynomial, arising from a determinant, is efficiently computable. Using this fact, Theorem 3 and the equivalence between sampling and counting, we can deduce the following result.

▶ **Corollary 4.** *There is an algorithm, which given a PSD matrix $L \in \mathbb{R}^{m\times m}$, a cost vector $c \in \mathbb{Z}^m$ and a cost value $C \in \mathbb{Z}$ samples a set S of cost $c(S) \leq C$ with probability proportional to $\det(L_{S,S})$. The running time of the algorithm is polynomial with respect to m and $\|c\|_1$.*

From the above one can derive efficient sampling algorithms for several classes of constraint families $\mathcal{C}$ which have *succinct descriptions.* Indeed, we establish counting and sampling algorithms for a general class of *linear families* of the form

$$\mathcal{C} = \{S \subseteq [m] : c_1(S) \in K_1, c_2(S) \in K_2, \ldots, c_p(S) \in K_p\} \tag{1}$$

where $c_1, c_2, \ldots, c_p \in \mathbb{Z}^m$ and $K_1, \ldots, K_p \subseteq \mathbb{Z}$. We prove the following

▶ **Corollary 5.** *There is an algorithm, which given a PSD matrix $L \in \mathbb{R}^{m\times m}$ and a description of a linear family $\mathcal{C}$ as in* (1)*, samples a set $S \in \mathcal{C}$ with probability proportional to $\det(L_{S,S})$. The running time of the algorithm is polynomial in m and $\prod_{j=1}^{p}\left(\|c_j\|_1 + 1\right)$.*

One particular class of families for which the above yields polynomial time sampling algorithms are partition families (families of bases of partition matroids) over constantly many parts (see Corollary 10). An important open problem that remains is to come up with even faster algorithms.

Another application of Theorem 3, which we present in Section 6, is to combinatorial sampling and counting problems. More precisely, we note that the indicator measure of bases of regular matroids has an efficiently computable generating polynomial; hence, we can solve their corresponding budgeted versions of counting and sampling problems.

One may ask if the dependence on $\|c\|_1$ in Theorem 3 can be improved. We prove that the answer to this question is no in a very strong sense. To state our hardness result, we introduce ECount – a natural variant of the BCount problem – in which the sum is over subsets of cost equal to a given value C instead of at most C (such a problem is no harder than BCount). We provide an approximation preserving reduction showing that ECount$[\mu, c, C]$ is at least as hard as computing *mixed discriminants* of tuples of positive semidefinite (PSD) matrices when c and C are given in binary, and can be exponentially large in magnitude. Recall that for a tuple of $m \times m$ PSD matrices $A_1, \ldots, A_m$, their mixed discriminant is the coefficient of the monomial $\prod_{i=1}^m x_i$ in the polynomial $\det(\sum_{i=1}^m x_i A_i)$.

▶ **Theorem 6** (Hardness of Counting under Budget Constraints). BCount$[\mu, c, C]$ *is* $\#\mathbf{P}$*−hard. Moreover, when μ is a determinantal function,* ECount$[\mu, c, C]$ *is at least as hard to approximate as mixed discriminants of tuples of PSD matrices.*

To prove this result we show an *equivalence* between the counting problem corresponding to partition-constrained DPPs (with a large, super-constant number of parts) and computing mixed discriminants. Unlike permanents [21], no efficient approximation scheme is known for estimating mixed discriminants and there is some evidence [15] that there may be none. To further understand to what extent g_μ is the cause of computational hardness, in Appendix A (see Theorem 20) we provide another hardness result; it considers a μ that is a 0/1 indicator function for spanning trees in a graph (with efficiently computable g_μ). We prove that

$\textsc{ECount}[\mu, c, C]$ is at least as hard to approximate as the number of perfect matchings in general (non-bipartite) graphs, which is another problem for which existence of an FPRAS is open.

Finally, this connection between partition-DPPs and mixed discriminants, along with our results to efficiently solve the counting problem for partition-DPPs with constantly many parts, gives us other applications of independent interest. 1) The ability to compute the top few coefficients of the *mixed characteristic polynomial* that arises in the proof of the Kadison-Singer problem; see Theorem 15. 2) The ability to compute in polynomial time, the mixed characteristic polynomial *exactly*, when the linear matrix subspace spanned by the input matrices has constant dimension; see Theorem 17 and Corollary 18.

1.2 Other Related Work

For sampling from k-DPPs there are exact polynomial time algorithms (see [20, 10, 24]). There is also recent work on faster approximate MCMC algorithms for sampling from various unconstrained discrete point processes (see [31, 1] and the references therein), and algorithms that are efficient for constrained DPPs under certain restrictions on the kernel and constraints (see [25] and the references therein). To the best of our knowledge, our result is the first efficient sampling algorithm that works for all kernels and for any constraint set with small description complexity. Recently, approximate algorithms for the counting and sampling were presented in [32]. On the practical side, diverse subset selection and DPPs arise in a variety of contexts such as structured prediction [30], recommender systems [14] and active learning [34], where the study of DPPs with additional constraints is of importance.

2 Counting with Budget Constraints

Proof of Theorem 3. Let us first consider the case in which the cost vector c is nonnegative, i.e., $c \in \mathbb{N}^m$. We introduce a new variable z and consider the polynomial

$$h(z) \stackrel{\text{def}}{=} g_\mu(z^{c_1}, z^{c_2}, \ldots, z^{c_m}).$$

Since $g_\mu(x_1, \ldots, x_m) = \sum_{S\subseteq[m]} \mu(S) \prod_{i\in S} x_i$, we have

$$h(z) = \sum_{S\subseteq[m]} \mu(S) \prod_{i\in S} z^{c_i} = \sum_{S\subseteq[m]} \mu(S) z^{c(S)} = \sum_{0\le d\le \|c\|_1} z^d \sum_{S:\ c(S)=d} \mu(S).$$

Hence, the coefficient of z^d in $h(z)$ is equal to the sum of $\mu(S)$ over all sets S such that $c(S) = d$. In particular, the output is the sum of coefficients over $d \le C$.

It remains to show how to compute the coefficients of h. Note that we do not have direct access to g_μ. However, we can evaluate $g_\mu(x)$ at any input $x \in \mathbb{R}^m$, which in turn allows us to compute $h(z)$ for any input $z \in \mathbb{R}$. Since $h(z)$ is a polynomial of degree at most $\|c\|_1$, in order to recover the coefficients of h, it suffices to evaluate it at $\|c\|_1 + 1$ inputs and perform interpolation. When using FFT, the total running time becomes:

$$(\|c\|_1 + 1) \cdot T_\mu + \widetilde{O}(\|c\|_1),$$

where T_μ is the running time of the evaluation oracle for g_μ.

In order to deal with the case in which c has negative entries, consider a modified version of h:

$$h(z) \stackrel{\text{def}}{=} z^{\|c\|_1} g_\mu(z^{c_1}, z^{c_2}, \ldots, z^{c_m}).$$

Clearly, $h(z)$ is a polynomial of degree at most $2 \cdot \|c\|_1$ whose coefficients encode the desired output. ◀

▶ Remark. Note that the bit complexity of the output of the proposed algorithm is polynomial in the input size since it is a result of solving a linear system with all the coefficients being polynomially bounded.

We also state a simple consequence of the above proof that is often convenient to work with.

▶ **Corollary 7.** *There is an algorithm that, given a vector $c \in \mathbb{Z}^m$, a value $C \in \mathbb{Z}$ and oracle access to g_μ computes the sum $\sum_{S\colon\, c(S)=C} \mu(S)$ in time polynomial with respect to m and $\|c\|_1$.*

In the above, note the equality $c(S) = C$ instead of $c(S) \leq C$ as in BCount.

3 Determinantal Point Processes

A Determinantal Point Process (DPP) is a probability distribution μ over subsets of $[m]$ defined with respect to a symmetric positive semidefinite matrix $L \in \mathbb{R}^{m\times m}$ by $\mu(S) \propto \det(L_{S,S})$; i.e.,

$$\mu(S) \overset{\text{def}}{=} \frac{\det(L_{S,S})}{\sum_{T\subseteq[m]} \det(L_{T,T})}.$$

We will often use a different matrix to represent the measure μ; let $V \in \mathbb{R}^{m\times n}$ be a matrix, such that $L = VV^\top$ (the Cholesky decomposition of L). Then, $\det(L_{S,S}) = \det(V_S V_S^\top)$.

An important open problem related to DPPs is the sampling problem under additional combinatorial constraints imposed on the ground set $[m]$. We prove that these problems are polynomial time solvable for succinct budget constraints, as in Theorem 3. We start by establishing the fact that generating polynomials for determinantal distributions are efficiently computable.

▶ **Fact 8.** *Let $L \in \mathbb{R}^{m\times m}$ be a PSD matrix with $L = VV^\top$ for some $V \in \mathbb{R}^{m\times n}$. If $\mu : 2^{[m]} \to \mathbb{R}_{\geq 0}$ is defined as $\mu(S) \overset{\text{def}}{=} \det(L_{S,S})$ then $\det(V^\top XV + I) = \sum_{S\subseteq[m]} x^S \mu(S)$, where X is the diagonal matrix of indeterminates $X = \mathrm{Diag}\,(x_1, \ldots, x_m)$ and I is the $n \times n$ identity matrix.*

Proof. We start by applying the Sylvester's determinant identity

$$\det(V^\top XV + I) = \det\left(\left(\sqrt{X}V\right)\left(\sqrt{X}V\right)^\top + I\right).$$

It is well known that for a symmetric matrix $A \in \mathbb{R}^{m\times n}$ the coefficient of t^k in the polynomial $\det(A + tI)$ is equal to $\sum_{|S|=n-k} \det(A_{S,S})$. Applying this result to $A = \left(\sqrt{X}V\right)\left(\sqrt{X}V\right)^\top$, we get

$$\det(A_{S,S}) = x^S \det(V_S V_S^\top) = x^S \det(L_{S,S}),$$

which concludes the proof by simply taking $t = 1$. ◀

Now we are ready to deduce Corollary 4.

Proof of Corollary 4. A polynomial time counting algorithm follows directly from Theorem 3 and Fact 8. To deduce sampling we apply the result on equivalence between sampling and counting Theorem 21. In fact when applied to an exact counting algorithm we obtain an exact sampling procedure. ◀

We move to the general result on sampling for linear families – Corollary 5. One can deduce it directly from Theorem 3, but this leads to a significantly suboptimal algorithm. Instead we take a different path and reprove Theorem 3 in a slightly higher generality.

Proof of Corollary 5. We will show how to solve the counting problem – sampling will then follow from Theorem 21. Also, for simplicity we assume that all the entries in the cost vectors are nonnegative, this can be extended to the general setting as in the proof of Theorem 3.

Let g be the generating polynomial of the determinantal function $\mu(S) = \det(L_{S,S})$, which is efficiently computable by Fact 8. For notational clarity we will use superscripts to index constraints. For every constraint "$c^{(j)}(S) \in K_i$" $(j = 1, 2, \dots, p)$ introduce a new formal variable y_j. For every index $i \in [m]$ define the monomial:

$$s_i = \prod_{j=1}^{p} y_j^{c_i^{(j)}}.$$

The above encodes the cost of element i with respect to all cost vectors $c^{(j)}$ for $j = 1, 2, \dots, p$. Consider the polynomial $h(y_1, \dots, y_p) = g(s_1, s_2, \dots, s_m)$. It is not hard to see that the coefficient of a given monomial $\prod_{j=1}^{p} y_j^{d_j}$ in h is simply the sum of $\mu(S)$ over all sets S satisfying $c^{(1)}(S) = d_1, c^{(2)}(S) = d_2, \dots, c^{(p)}(S) = d_p$. Hence the solution to our counting problem is simply the sum of certain coefficients of h. It remains to show how to recover all the coefficients efficiently.

Note that we can efficiently evaluate the polynomial h at every input $(y_1, \dots, y_p) \in \mathbb{R}^p$. One can then apply interpolation to recover all coefficients of h. The running time is polynomial in the total number of monomials in h (this is the number of variables of a linear system which can be used to find the coefficients), which can be bounded from above by $\prod_{j=1}^{p} \left(\left\|c^{(j)}\right\|_1 + 1\right)$. ◀

We derive now one interesting application of Corollary 5 – sampling from partition constrained DPPs. Let us first define *partition families* formally.

▶ **Definition 9.** Let $[m] = P_1 \cup P_2 \cup \dots \cup P_p$ be a partition of $[m]$ into disjoint, nonempty sets and let $b_1, b_2, \dots, b_p$ be integers such that $0 \le b_i \le |P_i|$. A family of sets of the form

$$\mathcal{C} = \{S \subseteq [m] : |S \cap P_j| = b_j, \text{ for every } j = 1, 2, \dots, p\}$$

is called a partition family.

We prove the following consequence of Corollary 5, which asserts that polynomial time counting and sampling is possible for DPPs under partition constraints for constant p.

▶ **Corollary 10.** *Given a DPP defined by $L \in \mathbb{R}^{m \times m}$ and a partition family $\mathcal{C}$ with a constant number of parts, there exists a polynomial time sampling algorithm for the distribution*

$$\mu_{\mathcal{C}}(S) \stackrel{\text{def}}{=} \frac{\det(L_{S,S})}{\sum_{T \in \mathcal{C}} \det(L_{T,T})} \qquad \text{for } S \in \mathcal{C}.$$

Proof. In light of Corollary 5 it suffices to show that every partition family has a succinct representation as a linear family. We show that it is indeed the case. Consider a partition family $\mathcal{C}$ induced by the partition $P_1 \cup P_2 \cup \ldots \cup P_p = [m]$ and numbers $b_1, b_2, \dots, b_p$. Define the following cost vectors: $c_j = 1_{P_j}$, for $j = 1, 2, \dots, p$, i.e., the indicator vectors of the sets $P_1, P_2, \dots, P_p$. Moreover define K_j to be $\{b_j\}$ for every $j = 1, 2, \dots, p$. It is then easy to see that "$c_j(S) \in K_j$" is implementing the constraint $|P_j \cap S| = b_j$. In other words the family $\mathcal{C}$ is equal to the linear family defined by cost vectors $c_1, c_2, \dots, c_p$ and sets $K_1, K_2, \dots, K_p$. It remains to observe that $\|c_j\|_1 = |P_j| \le m$ and hence $\prod_{j=1}^{p} (\|c_j\| + 1) = O(m^p)$. Since $p = O(1)$ the algorithm from Corollary 5 runs in polynomial time. ◀

4 Hardness Result

In this section we study hardness of BCOUNT$[\mu, c, C]$. Theorem 3 implies that BCOUNT is polynomial time solvable whenever we measure the complexity with respect to the unary encoding length of the cost vector c. Here we prove that if c is given in binary, the problem becomes $\#\mathbf{P}-$hard. Moreover, existence of an efficient approximation scheme for a closely related problem (instead of counting all objects of cost *at most* C, count objects of cost *exactly* C) would imply existence of such schemes for counting perfect matchings in non-bipartite graphs (see Appendix A) and for computing mixed discriminants. In both cases, these are notorious open questions and the latter is believed to be unlikely.

4.1 Mixed Discriminants

We relate the BCOUNT problem to the well studied problem of computing mixed discriminants of PSD matrices and prove Theorem 6. Recall the definition:

▶ **Definition 11.** Let $A_1, A_2, \ldots, A_m \in \mathbb{R}^{d\times d}$ be symmetric matrices of dimension d. The mixed discriminant of a tuple $(A_1, A_2, \ldots, A_d)$ is defined as

$$D(A_1, A_2, \ldots, A_d) \overset{\text{def}}{=} \frac{\partial^d}{\partial z_1 \ldots \partial z_d} \det(z_1 A_1 + z_2 A_2 + \cdots + z_d A_d).$$

Computing mixed discriminants of PSD matrices is known to be $\#\mathbf{P}$-hard, since they can encode the permanent. However, as opposed to the permanent, there is no FPRAS known for computing mixed discriminants, and the best polynomial time approximation algorithms by [4, 16] have an exponentially large approximation ratio.

The main technical component in our proof of Theorem 6 is the following lemma.

▶ **Lemma 12.** *There is a polynomial time reduction, which given a tuple $(A_1, \ldots, A_n)$ of PSD $n \times n$ matrices outputs a PSD matrix $L \in \mathbb{R}^{m\times m}$, a cost vector $c \in \mathbb{Z}^m$ and a cost value $C \in \mathbb{Z}$ such that*

$$n! \cdot D(A_1, A_2, \ldots, A_n) = \sum_{S\subseteq[m],\ c(S)=C} \mu(S),$$

where $\mu(S) = \det(L_{S,S})$, for $S \subseteq [m]$. Moreover, $\|c\|_1 \le 2^{O(n \log n)}$.

Before proving Lemma 12 let us first state several important properties of mixed discriminants, which we will rely on; for proofs of these facts we refer the reader to [3].

▶ **Fact 13** (Properties of Mixed Discriminants). *Let $A, B, A_1, A_2, \ldots, A_n$ be symmetric $n \times n$ matrices.*

1. *D is symmetric, i.e.,*

$$D(A_1, A_2, \ldots, A_n) = D(A_{\sigma(1)}, A_{\sigma(2)}, \ldots, A_{\sigma(n)}), \text{ for any permutation } \sigma \in S_n.$$

2. *D is linear with respect to every coordinate, i.e.,*

$$D(\alpha A + \beta B, A_2, \ldots, A_n) = \alpha D(A, A_2, \ldots, A_n) + \beta D(B, A_2, \ldots, A_n).$$

3. *If $A = \sum_{i=1}^n v_i v_i^\top \in \mathbb{R}^{n\times n}$ then we have: $\det(A) = n!\ D(v_1 v_1^\top, \ldots, v_n v_n^\top)$.*

Proof of Lemma 12. Consider a tuple $(A_1, A_2, \ldots, A_n)$ of PSD matrices. The first step is to decompose them into rank-one summands:

$$A_i = \sum_{j=1}^{r} v_{i,j} v_{i,j}^{\top},$$

where $v_{i,j} \in \mathbb{R}^n$ for $1 \le i, j \le n$ (some $v_{i,j}$'s can be zero if $\operatorname{rank}(A_i) < n$). This step can be performed using the Cholesky decomposition.

Let $M = \{(i,j) : 1 \le i, j \le n\}$ and for every $i = 1, 2, \ldots, n$ define $P_i = \{i\} \times [n]$. We take $m = |M| = n^2$ and define a family $\mathcal{C}$ of $n-$subsets of M to be

$$\mathcal{C} = \{S \subseteq [m] : |S \cap P_i| = 1 \text{ for every } i = 1, 2, \ldots, n\}.$$

Let V denote an $m \times n$ matrix with rows indexed by M, for which the eth row is v_e as above ($e \in M$, i.e., $e = (i,j)$ for some $i, j \in [n]$). We also set $L = VV^{\top}$, hence L is an $m \times m$ symmetric, PSD matrix. Finally, let $\mu(S) = \det(L_{S,S})$. Note that for sets S of cardinality n we have

$$\mu(S) = \det(L_{S,S}) = \det(V_S V_S^{\top}) = \det(V_S^{\top} V_S) = \det\left(\sum_{e \in S} v_e v_e^{\top}\right).$$

In the calculation below we rely on properties of mixed discriminants listed in Fact 13 and on the fact that $|S| = n$ for $S \in \mathcal{C}$.

$$\begin{aligned} D(A_1, A_2, \ldots, A_n) &= D\left(\sum_{j=1}^{n} v_{1,j} v_{1,j}^{\top}, \sum_{j=1}^{n} v_{2,j} v_{2,j}^{\top}, \ldots, \sum_{j=1}^{n} v_{n,j} v_{n,j}^{\top}\right) \\ &= \sum_{1 \le j_1, j_2, \ldots, j_n \le n} D(v_{1,j_1} v_{1,j_1}^{\top}, v_{2,j_2} v_{2,j_2}^{\top}, \ldots, v_{n,j_n} v_{n,j_n}^{\top}) \\ &= \sum_{e_1 \in P_1, e_2 \in P_2, \ldots, e_n \in P_n} D(v_{e_1} v_{e_1}^{\top}, v_{e_2} v_{e_2}^{\top}, \ldots, v_{e_n} v_{e_n}^{\top}) \\ &= \sum_{\{e_1, e_2, \ldots, e_n\} \in \mathcal{C}} \frac{1}{n!} \det(v_{e_1} v_{e_1}^{\top} + v_{e_2} v_{e_2}^{\top} + \ldots + v_{e_n} v_{e_n}^{\top}) = \frac{1}{n!} \sum_{S \in \mathcal{C}} \mu(S). \end{aligned}$$

It remains to show that the partition family $\mathcal{C}$ can be represented as $\mathcal{C} = \{S \subseteq M : c(S) = C\}$ for some cost vector $c \in \mathbb{Z}^M$ and $C \in \mathbb{Z}$, such that $\|c\|_1 = 2^{O(n \log n)}$. Indeed, by a reasoning as in Corollary 10 we can represent $\mathcal{C}$ as a linear family with n constraints of the form $c^{(i)}(S) = 1$ for $i = 1, 2, \ldots, n$ and $c^{(i)} \in \{0,1\}^{n \times n}$. It is not hard to see that these can be combined into one constraint $c(S) = C$ with $\|c\|_1 = (n^2)^{n+O(1)} = 2^{O(n \log n)}$. Now, it remains to observe that all the steps of the reduction are efficient (since the cost vector is represented in binary here). ◀

Proof of Theorem 6. In light of Lemma 12, the problem of computing $\sum_{S \subseteq [m], c(S) = C} \mu(S)$ for determinantal functions μ is at least as hard as computing mixed discriminants. The BCount problem is very similar, with the only difference that it is computing the sum over all sets of cost $c(S)$ at most C. However, clearly by solving the BCount problem for C and $C - 1$ one can compute $\sum_{S \subseteq [m], c(S) = C} \mu(S)$ by just subtracting the obtained results. ◀

5 Mixed Discriminants and Mixed Characteristic Polynomials

Mixed Characteristic Polynomials played a crucial role in the proof of the Kadison-Singer conjecture. Making this proof algorithmic is an outstanding open question that naturally leads to the problem of computing the maximum root of these mixed characteristic polynomials. In this section, we show how Corollary 10 implies a polynomial time algorithm for higher-order coefficients of such polynomials. We start by defining mixed characteristic polynomials. We use the following simplified notation for partial derivatives: $\partial_{x_i} f(x)$ is an abbreviation for $\frac{\partial}{\partial x_i} f(x)$.

▶ **Definition 14.** Let $A_1, A_2, \ldots, A_m \in \mathbb{R}^{d\times d}$ be symmetric matrices of dimension d. The mixed characteristic polynomial of $A_1, A_2, \ldots, A_m$ is defined as

$$\mu[A_1, \ldots, A_m](x) \overset{\text{def}}{=} \prod_{i=1}^{m} (1 - \partial_{z_i}) \det \left(xI + \sum_{i=1}^{m} z_i A_i \right) \Bigg|_{z_1=\cdots=z_m=0}.$$

Note in particular that while mixed discriminants are defined for a tuple whose length matches the dimension d of the matrices, for the case of mixed characteristic polynomials the number m can be arbitrary. In fact, when $m = d$, the constant term in the mixed characteristic polynomial is (up to sign) equal to the mixed discriminant of the input tuple.

However, one may wonder whether all of the coefficients in these polynomials are hard to compute. The following result shows that higher-degree coefficients are computable in polynomial time. Roughly, the proof relies on the observation that the higher-degree coefficients in the mixed characteristic polynomial are sums of mixed discriminants that only have constantly many *distinct* matrices. As we demonstrate, computing such mixed discriminants reduces to counting for DPPs under partition constraints with a constant number of parts, which allows us to apply Corollary 10. The formal statement of the theorem follows [2].

▶ **Theorem 15.** *Given a set of m symmetric, PSD matrices $A_1, \ldots, A_m \in \mathbb{R}^{d\times d}$, one can compute the coefficient of x^{d-k} in $\mu[A_1, \ldots, A_m](x)$, in* $\mathrm{poly}(m^k)$ *time.*

An important component in the proof of Theorem 15 is a reduction from counting for partition constrained DPPs to mixed discriminants. In fact we use it as a subroutine for computing higher-order coefficients of the mixed characteristic polynomial. In Section 4 we provided a reduction in the opposite direction, thus establishing an *equivalence* between mixed discriminants and counting for partition constrained DPPs.

▶ **Lemma 16.** *Given a set of m vectors $v_1, \ldots, v_m \in \mathbb{R}^r$ and a partition of $[m] = P_1 \cup \cdots \cup P_p$ into disjoint, non-empty sets, consider a partition family $\mathcal{C} = \{S \subseteq [m] : |S \cap P_j| = b_j$ for every $j = 1, 2, \ldots, p\}$ such that $\sum_{j=1}^{p} b_j = r$. Let $(A_1, \ldots, A_r)$ be an r-tuple of PSD $r \times r$ matrices such that $(A_1, A_2, \ldots, A_r) = (\overbrace{B_1, \ldots, B_1}^{b_1 \text{ times}}, \overbrace{B_2, \ldots, B_2}^{b_2 \text{ times}}, \ldots, \overbrace{B_p, \ldots, B_p}^{b_p \text{ times}})$ where $B_i = \sum\limits_{e\in P_i} v_e v_e^\top$ for every partition P_i, the following equality holds:*

$$\prod_{i=1}^{p} b_i! \cdot D(A_1, A_2, \ldots, A_r) = \sum_{S\in\mathcal{C}} \det(V_S V_S^\top),$$

where $V \in \mathbb{R}^{m\times r}$ denotes the matrix formed by arranging the vectors $v_1, \ldots, v_m$ row-wise.

Proof. Consider the quantities B_i and $(A_1, A_2, \ldots, A_r)$ as defined in the theorem. By applying linearity multiple times to all coordinates of $D(A_1, A_2, \ldots, A_r)$ we find that:

$$D(A_1, A_2, \ldots, A_r) = \alpha \sum_{S\in\mathcal{B}} D(v_{e_1} v_{e_1}^\top, v_{e_2} v_{e_2}^\top, \ldots, v_{e_r} v_{e_r}^\top),$$

[2] Independent of our work which first appeared in [9], a recent preprint [2] devise a different algorithm to obtain a similar result

where S is $\{e_1, e_2, \ldots, e_r\}$ in the summation above and α is $\prod_{i=1}^{p} b_i!$. This is because $D(v_{e_1}v_{e_1}^\top, v_{e_2}v_{e_2}^\top, \ldots, v_{e_r}v_{e_r}^\top) = 0$ whenever $e_1, e_2, \ldots, e_r$ are not pairwise distinct. We use Fact 13 again to obtain that

$$D(v_{e_1}v_{e_1}^\top, v_{e_2}v_{e_2}^\top, \ldots, v_{e_r}v_{e_r}^\top) = \frac{1}{r!}\det(v_{e_1}v_{e_1}^\top + v_{e_2}v_{e_2}^\top + \ldots + v_{e_r}v_{e_r}^\top) = \det(V_S V_S^\top).$$

This concludes the proof. Furthermore, it is evident that the r-tuple $(A_1, A_2, \ldots, A_r)$ is efficiently computable given the partition family $\mathcal{C}$ and matrix V. ◀

Proof of Theorem 15. First note that without loss of generality we can assume that $d \leq m$, as otherwise – if $d > m$ we can add $(d - m)$ zero-matrices which does not change the result but places us in the $d \leq m$ case. The starting point of our proof is an observation made in [27] which provides us with another expression for the mixed characteristic polynomial in terms of mixed discriminants:

$$\mu[A_1, \ldots, A_m](x) = \sum_{k=0}^{d} x^{d-k}(-1)^k \sum_{S \in \binom{[m]}{k}} D((A_i)_{i \in S}) \tag{2}$$

where we denote $D(A_1, \ldots, A_k) = \frac{1}{(d-k)!}D(A_1, \ldots, A_k, I, \ldots, I)$ with the identity matrix I repeated $d - k$ times. Therefore, our task reduces to computing $O(m^k)$ mixed discriminants of the form $D(A_1, \ldots, A_k, I, \ldots, I)$. Below we show that such a quantity is computable in poly(d^k) time which concludes the proof.

Consider the Cholesky decomposition of A_i for $i = 1, 2, \ldots, k+1$ (we set $A_{k+1} = I$ for convenience)

$$A_i = \sum_{j=1}^{d} u_{i,j}u_{i,j}^\top.$$

Let $M = \{(i, j) : 1 \leq i \leq k+1, 1 \leq j \leq d\}$ be the ground set of a partition family of size $m \overset{\text{def}}{=} (k+1)d$. Define an $m \times d$ matrix U by placing $u_{i,j}$'s as rows of U.

Further, consider a partition $M = P_1 \cup \cdots \cup P_{k+1}$ with $P_i = \{i\} \times [d]$ for all $i = 1, \ldots, k+1$ and let $b_1 = \ldots = b_k = 1$ and $b_{k+1} = d - k$. This gives rise to a partition family

$$\mathcal{C} = \{T \in M : |T \cap P_i| = b_i \text{ for all } i = 1, \ldots, k+1\}.$$

We claim that

$$\prod_{i=1}^{k+1} b_i! \sum_{T \in \mathcal{C}} \det(U_T U_T^\top) = D(A_1, \ldots, A_k, I \ldots, I). \tag{3}$$

This follows from Lemma 16 by considering this partition family $\mathcal{C}$ and matrix U as defined here. Equation (3) combined with the counting result for DPPs under partition constraints (Corollary 10) conclude the proof. ◀

The second observation is more general in its nature and tries to answer the question whether computing mixed characteristic polynomials is strictly harder than computing mixed discriminants. In fact, as noted above, the coefficients of mixed characteristic polynomials are expressed as sums of (an exponential number of) mixed discriminants. We show that these exponential sums can be computed by evaluating a *single* mixed discriminant of matrices of size at most $d+n$. Moreover, our reduction is *approximation-preserving*, hence demonstrating that approximating mixed discriminants are computationally equally hard as approximating the coefficients of the mixed characteristic polynomials. We remark that our reduction can be thought of as a generalization of a result for approximating the number of k-matchings in a bipartite graph ([13]).

▶ **Theorem 17.** *Given a tuple of m symmetric, positive semi-definite matrices $A_1, \ldots, A_m \in \mathbb{R}^{d\times d}$ with $d \le m$ and $k \in \{1, \ldots, d\}$, there exist a tuple of $m+d-k$ symmetric, positive semi-definite matrices $B_1, \ldots, B_{m+d-k} \in \mathbb{R}^{(m+d-k)\times(m+d-k)}$ such that the coefficient of x^{d-k} in the mixed characteristic polynomial $\mu[A_1, \ldots, A_m](x)$,*

$$\sum_{S\in\binom{[m]}{k}} D((A_i)_{i\in S}) = \frac{1}{(m-k)!(d-k)!} D(B_1, \ldots, B_{m+d-k})$$

Proof. We first show how to construct the $m+d-k$ matrices $B_1, \ldots, B_{m+d-k}$ from $A_1, \ldots, A_m$. The matrices $B_1, \ldots, B_{m+d-k}$ that we consider are 2-by-2 block diagonal matrices that we construct by taking appropriate direct sums. Recall that the direct sum of two matrices A and B of size $d_1 \times d_1$ and $d_2 \times d_2$ is a matrix of size $(d_1+d_2)\times(d_1+d_2)$ defined as

$$G = \left[\begin{array}{c|c} A & \mathbf{0}_{d_1\times d_2} \\ \hline \mathbf{0}_{d_2\times d_1} & B \end{array}\right]$$

where $\mathbf{0}_{m\times n}$ is an m-times-n matrix consisting of all zeros. We define the first m matrices to be direct sums of the A_i matrices with the identity matrix of order $m-k$, i.e., I_{m-k} and the remaining $d-k$ matrices to all be equal to the direct sum of the identity matrix of order d, i.e., I_d with the square zero matrix of order $m-k$, i.e., $\mathbf{0}_{m-k}$. Formally,

$$B_i = \begin{cases} A_i \oplus I_{m-k} & \text{for } i \in \{1, \ldots, k\}, \\ I_d \oplus \mathbf{0}_{m-k} & \text{otherwise} \end{cases}$$

We now proceed to prove the claim of the theorem from the definition of the mixed discriminant in Definition 14. For any subset $S \subseteq [m]$, denote $\partial^S = \prod_{i\in S} \partial_{z_i}$.

$$\begin{aligned}
&D(B_1, \ldots, B_{m+d-k}) \\
&= \partial_{z_1} \ldots \partial_{z_{m+d-k}} \det(z_1B_1 + \ldots + z_{m+d-k}B_{m+d-k}) \\
&= \partial_{z_1} \ldots \partial_{z_{m+d-k}} \det \left[\begin{array}{c|c} \sum_{i=1}^{m} z_iA_i + \sum_{i=1}^{d-k} z_{m+i}I_d & \mathbf{0}_{d\times(m-k)} \\ \hline \mathbf{0}_{(m-k)\times d} & \sum_{i=1}^{m} z_iI_{m-k} \end{array}\right] \\
&= \partial_{z_1} \ldots \partial_{z_{m+d-k}} (z_1 + \ldots + z_m)^{m-k} \det\left(\sum_{i=1}^{m} z_iA_i + \sum_{i=1}^{d-k} z_{m+i}I_d\right) \\
&= \sum_{\substack{S\subseteq[m] \\ |S|=m-k}} \left[\partial^S (z_1 + \ldots + z_m)^{m-k}\right] \left[\partial^{S^c} \prod_{i=1}^{d-k} \partial_{z_{m+i}} \det\left(\sum_{i=1}^{m} z_iA_i + \sum_{i=1}^{d-k} z_{m+i}I_d\right)\right] \\
&= \sum_{\substack{S\subseteq[m] \\ |S|=m-k}} (m-k)!\partial^{S^c} \prod_{i=1}^{d-k} \partial_{z_{m+i}} \det(\sum_{i\in S^c} z_iA_i + (z_{m+1} + \ldots z_{m+d-k})I_d)
\end{aligned}$$

$$= (m-k)! \sum_{\substack{S\subseteq[m]\\|S|=k}} D((A_i)_{i\in S}, \overbrace{I,\ldots,I}^{d-k \text{ times}})$$

$$= (m-k)!(d-k)! \sum_{\substack{S\subseteq[m]\\|S|=k}} D((A_i)_{i\in S})$$

The fourth to last equality follows simply from chain rule. Since we have an equality in the expression, the reduction is clearly approximation preserving and we are done. ◀

The above theorem in particular allows us to compute in polynomial time, the mixed characteristic polynomial *exactly*, when the linear matrix subspace spanned by the input matrices has constant dimension. This follows by combining Theorem 17 with Theorem 5.1 in [15].

▶ **Corollary 18.** *Suppose $A_1, A_2, \ldots, A_m \in \mathbb{R}^{d\times d}$ span a linear space of dimension k, then there exists a deterministic algorithm to compute $\mu[A_1,\ldots,A_m](x)$ in $\mathrm{poly}(m^k)$ time.*

Proof. In the proof of Theorem 17, the mixed discriminants computed are not of $A_1,\ldots,A_m$ but rather are of modified matrices. However, it is easy to see that for all tuples on which mixed discriminant is called, the dimension of the linear space spanned by them is at most $k+1$. It is proved in [15] that such mixed discriminants can be computed in $O(m^{2k+2})$ time. ◀

6 Budget-Constrained Sampling and Counting for Regular Matroids

Consider the following problem: given an undirected graph G with weights $c \in \mathbb{R}^m$ on its edges, sample a uniformly random spanning tree of cost at most C in G. This generalizes the problem of sampling uniformly random spanning trees [29] and sampling a random spanning tree of minimum cost [12]. Below we study the generalized version of this problem by considering regular matroids, indeed spanning trees arise as bases of the graphic matroid, which is known to be regular. We prove that the counting and sampling problem in this setting can be solved efficiently whenever c is polynomially bounded.

▶ **Theorem 19** (Counting and Sampling Bases of Matroids)**.** *Let $\mathcal{M}$ be a regular matroid on a ground set $[m]$ with a set of bases $\mathcal{B}$. There exists a counting algorithm which, given a cost vector $c \in \mathbb{Z}^m$ and a value $C \in \mathbb{Z}$, outputs the cardinality of the set $\{S \in \mathcal{B} : c(S) \le C\}$ and a sampling algorithm which, given a cost vector $c \in \mathbb{Z}^m$ and a value $C \in \mathbb{Z}$, outputs a random element in the set $\{S \in \mathcal{B} : c(S) \le C\}$. The running time of both algorithms is polynomial in m and $\|c\|_1$.*

Proof of Theorem 19. Let $\mathcal{M} \subseteq 2^{[m]}$ be a regular matroid and $\mathcal{B} \subseteq 2^{[m]}$ be its set of bases. We prove that the generating polynomial $\sum_{S\in\mathcal{B}} x^S$ is efficiently computable. We use the characterization of regular matroids as those which can be linearly represented by a totally unimodular matrix. In other words, there exists a totally unimodular matrix $A \in \mathbb{Z}^{m\times d}$ such that if we denote by $A_e \in \mathbb{Z}^d$ the e^{th} row of A it holds that:

$$S \in \mathcal{M} \quad \Leftrightarrow \quad \{A_e : e \in S\} \text{ is linearly independent.} \tag{4}$$

Let $r \le d$ be the rank of the matroid $\mathcal{M}$, i.e., the cardinality of any set in $\mathcal{B}$. We claim that without loss of generality one can assume that $d = r$. Indeed, we prove that there is a submatrix $A' \in \mathbb{Z}^{m\times r}$ of A, such that (4) still holds with A replaced by A'. To this end suppose that $d > r$. It is easy to see that the rank of A is r, otherwise, by (4) there would be a set S of cardinality at least $r+1$ with $S \in \mathcal{M}$. Hence there is a column in A which is a linear combination of the remaining columns, we can freely remove this column from A, while (4) will be still true. By doing so, we finally obtain a matrix A' with exactly r rows, which satisfies (4).

By the fact that A has r columns we have:

$$S \in \mathcal{B} \quad \Leftrightarrow \quad A_S \text{ is nonsingular,} \tag{5}$$

where by A_S we mean the $|S| \times r$ submatrix of A corresponding to rows from S. In particular, for a set $S \subseteq [m]$ of cardinality r we have:

$$S \in \mathcal{B} \quad \Leftrightarrow \quad \det(A_S) \neq 0 \quad \Leftrightarrow \quad \det(A_S^\top A_S) = 1, \tag{6}$$

where the last equivalence follows from A being totally unimodular. Let us now consider the polynomial

$$g(x_1, x_2, \ldots, x_m) = \det\left(\sum_{e=1}^{m} x_e A_e A_e^\top\right).$$

By the Cauchy-Binet theorem we obtain:

$$g(x_1, x_2, \ldots, x_m) = \sum_{|S|=r} \det\left(\sum_{e\in S} x_e A_e A_e^\top\right) = x^S \det(A_S^\top A_S).$$

In other words, g is equal to g_μ – the generating polynomial of the function $\mu : 2^{[m]} \to \mathbb{R}$ given by

$$\mu(S) = \begin{cases} 1 & \text{if } S \in \mathcal{B} \\ 0 & \text{otherwise.} \end{cases}$$

Therefore, since g_μ is efficiently computable, by Theorem 3 the BCount$[\mu, c, C]$ is efficiently solvable. This fact, together with Theorem 21 imply that sampling also can be made efficient. ◀

References

1 N. Anari, S. O. Gharan, and A. Rezaei. Monte Carlo Markov Chain Algorithms for Sampling Strongly Rayleigh Distributions and Determinantal Point Processes. In *COLT*, pages 103–115, 2016.

2 N. Anari, S. O. Gharan, A. Saberi, and N. Srivastava. Approximating the largest root and applications to interlacing families. *CoRR*, abs/1704.03892, 2017. URL: `http://arxiv.org/abs/1704.03892`.

3 R. B. Bapat. Mixed discriminants of positive semidefinite matrices. *Lin. Algebra & Applications*, 126, 1989.

4 A. Barvinok. Computing mixed discriminants, mixed volumes, and permanents. *Discrete & Computational Geometry*, 18, 1997.

5 D. Bertsimas and S. Vempala. Solving convex programs by random walks. *J. ACM*, July 2004.

6 A. Z. Broder and E. W. Mayr. Counting minimum weight spanning trees. *J. of Algorithms*, 24(1), 1997.

7 L. E. Celis, A. Deshpande, T. Kathuria, and N. K. Vishnoi. How to be fair and diverse? *Fairness, Accountability, and Transparency in Machine Learning*, 2016.

8 C. Chekuri, J. Vondrak, and R. Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. In *FOCS*, 2010.

9 A. Deshpande, T. Kathuria, D. Straszak, and N. K. Vishnoi. Combinatorial Determinantal Point Processes. *ArXiv e-prints*, 2016. arXiv:1608.00554.

10 A. Deshpande and L. Rademacher. Efficient Volume Sampling for Row/Column Subset Selection. In *FOCS*, Oct 2010.

11 M. E. Dyer, A. M. Frieze, and R. Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. *J. ACM*, 38(1):1–17, 1991.

12 D. Eppstein. *Representing all minimum spanning trees with applications to counting and generation.* UC Irvine, 1995.

13 S. Friedland and D. Levy. A polynomial-time approximation algorithm for the number of k-matchings in bipartite graphs. *ArXiv*, 2006. arXiv:0607135.

14 M. Gartrell, U. Paquet, and N. Koenigstein. Bayesian low-rank determinantal point processes. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*, pages 349–356, 2016.

15 L. Gurvits. On the complexity of mixed discriminants and related problems. In *MFCS*, 2005.

16 L. Gurvits and A. Samorodnitsky. A deterministic algorithm for approximating the mixed discriminant and mixed volume, and a combinatorial corollary. *Discrete & Computational Geometry*, 27(4), 2002.

17 B. Hajek. Cooling schedules for optimal annealing. *Mathematics of operations research*, 13(2), 1988.

18 N. Harvey. An introduction to the Kadison-Singer problem and the paving conjecture, 2013.

19 N. Harvey and N. Olver. Pipage rounding, pessimistic estimators and matrix concentration. In *SODA'14*, pages 926–945, 2014.

20 J. B. Hough, M. Krishnapur, Y. Peres, and B. Virág. Determinantal Processes and Independence. *ArXiv Mathematics e-prints*, March 2005. arXiv:math/0503110.

21 M. Jerrum, A. Sinclair, and E. Vigoda. A Polynomial-time Approximation Algorithm for the Permanent of a Matrix with Nonnegative Entries. *J. ACM*, 51(4), July 2004.

22 M.R. Jerrum, L.G. Valiant, and V.V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986.

23 A. Krause, A. Singh, and C. Guestrin. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *J. Mach. Learn. Res.*, 9, June 2008.

24 A. Kulesza and B. Taskar. Determinantal point processes for machine learning. *ArXiv*, July 2012. arXiv:1207.6083.

25 C. Li, S. Jegelka, and S. Sra. Markov chain sampling in discrete probabilistic models with constraints. In *NIPS*, 2016.

26 H. Lin and J. Bilmes. A Class of Submodular Functions for Document Summarization. In *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, HLT'11, 2011.

27 A. W. Marcus, D. A. Spielman, and N. Srivastava. Interlacing families. II: Mixed characteristic polynomials and the Kadison-Singer problem. *Ann. Math. (2)*, 182(1):327–350, 2015.

28 M. Mezard and A. Montanari. *Information, physics, and computation.* Oxford University Press, 2009.

29 R. Pemantle. Uniform random spanning trees. *arXiv preprint math/0404099*, 2004.

30 A. Prasad, S. Jegelka, and D. Batra. Submodular meets structured: Finding diverse subsets in exponentially-large structured item sets. In *Advances in Neural Information Processing Systems, December 8-13 2014, Montreal, Quebec, Canada*, pages 2645–2653, 2014.

31 P. Rebeschini and A. Karbasi. Fast mixing for discrete point processes. In *Proceedings of The 28th Conference on Learning Theory, COLT 2015, Paris, France, July 3-6, 2015*, pages 1480–1500, 2015.

32 Damian Straszak and Nisheeth K. Vishnoi. Real stable polynomials and matroids: Optimization and counting. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing*. ACM, 2017.

33 M. J. Wainwright, M. I. Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

34 K. Wei, R. K. Iyer, and J. A. Bilmes. Submodularity in data subset selection and active learning. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 1954–1963, 2015.

35 Y. Yue and T. Joachims. Predicting Diverse Subsets Using Structural SVMs. In *ICML*, 2008.

36 C. X. Zhai, W. W. Cohen, and J. Lafferty. Beyond Independent Relevance: Methods and Evaluation Metrics for Subtopic Retrieval. In *SIGIR*, 2003.

37 T. Zhou, Z. Kuscsik, J. Liu, M. Medo, J. R. Wakeling, and Y. Zhang. Solving the apparent diversity-accuracy dilemma of recommender systems. *PNAS*, 107(10), 2010.

A Hardness for Spanning Trees

We show that BCount is at least as hard as counting perfect matchings in a non-bipartite graph. The proof relies on a combinatorial reduction from counting perfect matchings in a graph to counting budget constrained spanning trees.

▶ **Theorem 20.** *There is a polynomial time reduction which given a graph $G = (V, E)$ with n vertices and m edges outputs a graph G' with n vertices and $O(m + n^2)$ edges, a cost vector $c \in \mathbb{N}^m$ with $\|c\|_1 \leq 2^{O(m \log m)}$ and a value $C \in \mathbb{N}$, such that:*

$$PM(G) = \alpha \cdot ST_C(G')$$

where $PM(G)$ denotes the number of perfect matchings in G, $ST_C(G')$ denotes the number of spanning trees of total cost C in G' and $\alpha = \frac{n^2}{2}(2n)^{-n/2}$.

Proof. Let $G = (V, E)$ be an undirected graph, let $n = |V|$ and $m = |E|$. We construct a new graph G' and a cost vector c, such that counting perfect matchings in G is equivalent to counting spanning trees of specified cost C in G' .

The graph $G' = (V, E')$ is obtained by adding a complete graph to G, i.e., $\binom{n}{2}$ edges, one between every pair of vertices. We call the set of new edges F, hence $E' = E \cup F$. Note that E' is a multiset. To all edges $e \in F$ we assign cost $c_e = 0$, while for the original edges the costs are positive and defined below.

Let $b = m' + 1$, where $m' = |E'|$ is the number of edges in G'. We define the cost of an edge $e = ij \in E$ to be:

$$c_e = b^i + b^j.$$

Note that from the choice of b and c it follows that given a cost $c(S)$ of some set $S \subseteq E$, we can exactly compute how many times a given vertex appears as an endpoint of an edge in S. Indeed, if we have:

$$c(S) = \sum_{i=1}^{n} \delta_i b^i$$

such that $0 \leq \delta_i \leq b-1$ (the $b-$ary representation of $c(S)$), then the degree of vertex i in S is δ_i. This follows from the fact that b is chosen to avoid carry overs when computing $c(S)$ in the $b-$ary numerical system. Therefore, it is now a natural choice to define $C \stackrel{\text{def}}{=} \sum_{i=1}^{n} b^i$. We claim that every perfect matching in G corresponds to exactly $\alpha = \frac{n^2}{2}(2n)^{-n/2}$ different spanning trees of cost C in G'.

To prove this claim, fix any spanning tree S of cost $c(S) = C$. Note first that we have $c(S \cap E) = c(S)$ because all of the edges $e \notin E$ have cost 0. Moreover, the set $M \stackrel{\text{def}}{=} S \cap E$ is a perfect matching in G, because $c(M) = C$ implies that the degree of every vertex in M is one. It remains to show that every perfect matching M in G corresponds to exactly α spanning trees of cost C in G.

Fix any perfect matching M_0 in G. We need to calculate how many ways are there to add $\frac{n}{2} - 1$ edges from E' to obtain a spanning tree of G'. By contracting the matching M_0 to $\frac{n}{2}$ vertices and considering edges in E' only, we obtain a complete graph on $\frac{n}{2}$ vertices with 4 parallel edges going between every pair of vertices. The answer is the number of spanning trees of the obtained graph. Cayley's formula easily implies that this number is $4^{\frac{n}{2}-1} \left(\frac{n}{2}\right)^{\frac{n}{2}-2}$ which equals α^{-1}. ◀

B Equivalence Between Counting and Sampling

In this section we state and prove a theorem that implies that the COUNT$[\mu, \mathcal{C}]$ and SAMPLE$[\mu, \mathcal{C}]$ problems are essentially equivalent. We prove that, for a given type of constraints $\mathcal{C}$, a polynomial time algorithm for counting can be transformed into a polynomial time algorithm for sampling and vice versa. This section follows the convention that $\mu : 2^{[m]} \to \mathbb{R}_{\geq 0}$ is any function that assigns nonnegative values to subsets of $[m]$ and $\mathcal{C} \subseteq 2^{[m]}$ is any family of subsets of $[m]$.

▶ **Theorem 21** (Equivalence Between Approximate Counting and Approximate Sampling). *Consider any function $\mu : 2^{[m]} \to \mathbb{R}_{\geq 0}$ and a family $\mathcal{C}$ of subsets of $[m]$. Let $\mu_{\mathcal{C}} : \mathcal{C} \to [0,1]$ be a distribution over $S \in \mathcal{C}$ such that $\mu_{\mathcal{C}}(S) \propto \mu(S)$. We assume evaluation oracle access to the generating polynomial g_μ of μ, and define the following two problems:*

- **Approximate $\mathcal{C}$-sampling:** *given a precision parameter $\varepsilon > 0$, provide a sample S from a distribution $\rho : \mathcal{C} \to [0,1]$ such that $\|\mu_{\mathcal{C}} - \rho\|_1 < \varepsilon$.*
- **Approximate $\mathcal{C}$-counting:** *given a precision parameter $\varepsilon > 0$, output a number $X \in \mathbb{R}$ such that $X(1+\varepsilon)^{-1} \leq \sum_{S \in \mathcal{C}} \mu(S) \leq X(1+\varepsilon)$.*

The time complexities of the above problems differ by at most a multiplicative factor of $\text{poly}(m, \varepsilon^{-1})$.

▶ Remark. Note that the above theorem establishes equivalence between *approximate* variants of COUNT$[\mu, \mathcal{C}]$ and SAMPLE$[\mu, \mathcal{C}]$. This is convenient for applications, because the exact counting variants of these problems are often #**P**−hard. Still, for some of them, efficient approximation schemes are likely to exist. Further, we mention that the implication from exact counting to exact sampling holds, hence the sampling algorithms that we obtain in this paper are exact.

Theorem 21 follows from a self-reducibility property [22] of the counting problem. Before we present the proof of Theorem 21, we introduce some terminology and state assumptions for the remaining part of this section. The function $\mu : 2^{[m]} \to \mathbb{R}_{\geq 0}$ is given as an evaluation oracle for $g_\mu(x) = \sum_{S \subseteq [m]} \mu(S) x^S$. In particular, we measure complexity with respect to the number of calls to such an oracle. An algorithm which, for a fixed family $\mathcal{C} \subseteq 2^{[m]}$ and every function μ, given access to g_μ computes $\sum_{S\in\mathcal{C}}\mu(S)$ is called a $\mathcal{C}$-counting oracle. Similarly, we define a $\mathcal{C}$-sampling oracle to be an algorithm which, given access to g_μ, provides samples from the distribution

$$\mu_{\mathcal{C}}(S) \stackrel{\text{def}}{=} \frac{\mu(S)}{\sum_{T\in\mathcal{C}}\mu(T)} \qquad \text{for } S\in\mathcal{C}.$$

B.1 Counting Implies Sampling

We now show how counting implies sampling. It proceeds by inductively conditioning on certain elements not being in the sample. For this idea to work one has to implement conditioning using the $\mathcal{C}-$sampling oracle and access to the generating polynomial only. Below we state the implication from counting to sampling in the exact variant. The approximate variant also holds, with an analogous proof.

▶ **Lemma 22** (Counting Implies Sampling). *Let $\mathcal{C}$ denote a family of subsets of $[m]$. Suppose access to a $\mathcal{C}$-counting oracle is given. Then, there exists a $\mathcal{C}$-sampling oracle which, for any function $\mu : 2^{[m]} \to \mathbb{R}_{\geq 0}$, makes* poly$(m)$ *calls to the counting oracle and to g_μ and outputs a sample from the distribution $\mu_{\mathcal{C}}$.*

Proof. Let $\mathbf{S}$ be the random variable corresponding to the sample our algorithm outputs; our goal is to have $\mathbf{S}\sim\mu_{\mathcal{C}}$. The sampling algorithm proceeds as follows: It sequentially considers each element $e\in[m]$ and tries to decide (at random) whether to include $e\in\mathbf{S}$ or not. To do so, it first computes the probability $\mathbb{P}(e\in\mathbf{S})$ *conditioned on all decisions thus far.* It then flips a biased coin with this probability, and includes e in $\mathbf{S}$ according to its outcome. More formally, the sampling algorithm can be described as follows:

1. Input: $V\in\mathbb{R}^{m\times r}$, a number $k\leq r$.
2. Initialize: $Y=\emptyset$, $N=\emptyset$.
3. For $e=1,2,\ldots,m$:
   - a. Compute the probability $p=\mathbb{P}(e\in\mathbf{S} : Y\subseteq\mathbf{S}, N\cap\mathbf{S}=\emptyset)$ under the distribution $\mathbf{S}\sim\mu_{\mathcal{C}}$.
   - b. Toss a biased coin with success probability p. In case of success add e to the set Y, otherwise add e to N.
4. Output: $\mathbf{S}=Y$.

It is clear that the above algorithm correctly samples from $\mu_{\mathcal{C}}$. It remains to show that $\mathbb{P}(e\in S : Y\subseteq S, N\cap S=\emptyset)$ can be computed efficiently. This follows from Lemma 23 below. ◀

▶ **Lemma 23.** *Let Y and N be disjoint subsets of $[m]$ and consider any $e\in[m]$. Suppose $\mathbf{S}$ is distributed according to $\mu_{\mathcal{C}}$. If we are given access to a $\mathcal{C}$-counting oracle and to g_μ, then $\mathbb{P}(e\in\mathbf{S} : Y\subseteq\mathbf{S}, N\cap\mathbf{S}=\emptyset)$ can be computed in* poly(m) *time.*

Proof. Assume $e\in[m]\setminus(Y\cup N)$; otherwise the probability is clearly 0 or 1. Let $Y'=Y\cup\{e\}$, then

$$\mathbb{P}(e\in\mathbf{S} : Y\subseteq\mathbf{S}, N\cap\mathbf{S}=\emptyset) = \frac{\sum_{S\in\mathcal{C}, Y'\subseteq S, N\cap S=\emptyset}\mu(S)}{\sum_{S\in\mathcal{C}, Y\subseteq S, N\cap S=\emptyset}\mu(S)}.$$

We now show how to compute such sums: Introduce a new variable y, and for every $e\in[m]$ define:

$$w_e \stackrel{\text{def}}{=} \begin{cases} yx_e & \text{for } e\in Y,\\ 0 & \text{for } e\in N,\\ x_e & \text{otherwise.}\end{cases}$$

We interpret the expression $g_\mu(w_1, w_2, \ldots, w_m)$ as a generating polynomial for a certain function $\mu'(y) : 2^{[m]} \to \mathbb{R}$; i.e.,

$$g_{\mu'}(x) \overset{\text{def}}{=} g_\mu(w_1, w_2, \ldots, w_m) = \sum_{S \cap N = \emptyset} y^{|S \cap Y|} x^S \mu(S).$$

Define a polynomial

$$h(y) \overset{\text{def}}{=} \sum_{S \in \mathcal{C}, S \cap N = \emptyset} y^{|S \cap Y|} \mu(S).$$

It follows that $h(y)$ is a polynomial of degree at most $|Y|$. In fact, the sum we are interested in is simply the coefficient of $y^{|Y|}$ in $h(y)$. The last thing to note is that we can compute $h(y)$ exactly by evaluating it for $|Y| + 1$ different values of y and then performing interpolation. Hence, we just need to query the $\mathcal{C}$-counting oracle $(|Y| + 1)$ times giving it μ' as input (for various choices of y).[3] ◀

B.2 Sampling Implies Counting

We show the implication from sampling to counting in Theorem 21. Similarly as for the opposite direction we assume for simplicity that the sampling algorithm is exact, i.e., we prove the following lemma. The approximate variant holds with an analogous proof.

▶ **Lemma 24** (Sampling Implies Counting). *Let $\mathcal{C}$ denote a family of subsets of $[m]$. Suppose we have access to a $\mathcal{C}$-sampling oracle. Then, there exists a $\mathcal{C}$-counting oracle which for any input function $\mu : 2^{[m]} \to \mathbb{R}$ (given as an evaluation oracle for g_μ) and for any precision parameter $\varepsilon > 0$ makes* $\text{poly}(m, 1/\varepsilon)$ *calls to the sampling oracle, and approximates the sum:*

$$\sum_{S \in \mathcal{C}} \mu(S)$$

within a multiplicative factor of $(1 + \varepsilon)$. The algorithm has failure probability exponentially small in m.

Let us first state the algorithm which we use to solve the counting problem. Later in a sequence of lemmas we explain how to implement it in polynomial time and reason about its correctness. In the description, $\mathbf{S}$ denotes a random variable distributed according to $\mu_{\mathcal{C}}$.

```
1. Initialize $U \overset{\text{def}}{=} [m]$, $X \overset{\text{def}}{=} 1$.
2. Repeat
   a. Estimate the probability $\mathbb{P}(\mathbf{S} = U : \mathbf{S} \subseteq U)$, if it is larger than $(1 - \frac{1}{m})$, terminate the loop.
   b. Find an element $e \in U$ so that $\mathbb{P}(e \notin \mathbf{S} : \mathbf{S} \subseteq U) \geq \frac{1}{m^2}$.
   c. Approximate $p_e \overset{\text{def}}{=} \mathbb{P}(e \notin \mathbf{S} : \mathbf{S} \subseteq U)$ up to a multiplicative factor $\frac{\varepsilon}{m}$.
   d. Update $X \overset{\text{def}}{=} X \cdot \rho_e$, where $\rho_e$ is the estimate for $p_e$.
   e. Remove $e$ from $U$, i.e., set $U \overset{\text{def}}{=} U \setminus \{e\}$.
3. Return $X \cdot \mu(U)$.
```

[3] The provided argument does not generalize directly to the case when the counting oracle is only approximate (because of the interpolation step). However, as we need to compute the top coefficient of a polynomial $h(y)$ only, we can alternatively do it by evaluating $h(y)$ and dividing by y^d (for $d = \deg(h)$) at a very large input $y \in \mathbb{R}$.

▶ **Lemma 25.** *Given $U \subseteq [m]$ and $e \in U$, assuming access to a $\mathcal{C}$-sampling oracle, we can approximate the quantity*

$$p_e = \mathbb{P}(e \notin \mathbf{S} : \mathbf{S} \subseteq U)$$

where $\mathbf{S}$ is distributed according to $\mu_{\mathcal{C}}$, up to an additive error $\delta > 0$ in time $\frac{\mathrm{poly}(m)}{\delta^2}$. The probability of failure can be made $\frac{1}{m^c}$ for any $c > 0$.

Proof. We sample a set $S \in \mathcal{C}$ from the distribution $\mathbb{P}(S) \propto \mu(S)$ conditioned on $S \subseteq U$. This can be done using the sampling oracle, however instead of sampling with respect to μ one has to sample with respect to a modified function μ' which is defined as $\mu'(S) = \mu(S)$ for $S \subseteq U$ and $\mu'(S) = 0$ otherwise. Note that the generating polynomial for μ' can be easily obtained from g_μ by just plugging in zeros at positions outside of U. Given a sample S from μ' we define

$$X = \begin{cases} 1 & \text{if } e \notin S, \\ 0 & \text{otherwise.} \end{cases}$$

Repeat the above independently N times, to obtain $X_1, X_2, \ldots, X_N$ and finally compute the estimator:

$$Z = \frac{X_1 + X_2 + \cdots + X_N}{N}.$$

By Chebyshev's inequality, we have:

$$\mathbb{P}(|Z - p_e| \geq \delta) \leq \frac{1}{N\delta^2}.$$

Thus, by taking $N = \frac{\mathrm{poly}(m)}{\delta^2}$ samples, with probability $\geq 1 - \frac{1}{\mathrm{poly}(m)}$ we can obtain an additive error of at most δ. ◀

▶ **Lemma 26.** *If $U \subseteq [m]$ is such that $\mathbb{P}(\mathbf{S} = U : \mathbf{S} \subseteq U) \leq (1 - \frac{1}{m})$ then there exists an element $e \in U$ such that $\mathbb{P}(e \notin \mathbf{S} : \mathbf{S} \subseteq U) \geq \frac{1}{m^2}$, where $\mathbf{S}$ is distributed according to $\mu_{\mathcal{C}}$.*

Proof. Let $\mathbf{T}$ be the random variable $\mathbf{S}$ conditioned on $\mathbf{S} \subseteq U$. Denote $q_e = \mathbb{P}(e \in \mathbf{S} : \mathbf{S} \subseteq U)$, we obtain

$$\sum_{e \in U} q_e = \mathbb{E}(|\mathbf{T}|) \leq \left(1 - \frac{1}{m}\right)|U| + \frac{1}{m}(|U| - 1) = |U| - \frac{1}{m}.$$

The inequality in the above expression follows from the fact that the worst case upper bound would be achieved when the probability of $|\mathbf{T}| = |U|$ is *exactly* $1 - \frac{1}{m}$ and with the remaining probability, $|\mathbf{T}| = |U| - 1$. Hence $\sum_{e \in U}(1 - q_e) \geq \frac{1}{m}$, which implies that $(1 - q_e) \geq \frac{1}{m^2}$ for some $e \in U$. ◀

We are now ready to prove Lemma 24.

Proof of Lemma 24. We have to show that the algorithm given above can be implemented in polynomial time and it gives a correct answer.

Step 2(a) can be easily implemented by taking $\mathrm{poly}(m)$ samples conditioned on $\mathbf{S} \subseteq U$ (as in the proof of Lemma 25). This gives us an approximation of $q_U = \mathbb{P}(\mathbf{S} = U : \mathbf{S} \subseteq U)$

up to an additive error of at most m^{-2} with high probability. If the estimate is less than $(1-\frac{1}{2m})$ then with high probability $q_U \leq (1-\frac{1}{m})$ otherwise, with high probability we have

$$\mu(U) \leq \sum_{S\in\mathcal{C},S\subseteq U} \mu(S) \leq \left(1+\frac{4}{m}\right)\mu(U) \tag{7}$$

and the algorithm terminates.

When performing step 2(b) we have a high probability guarantee for the assumption of Lemma 26 to be satisfied. Hence, we can assume that (by using Lemma 26 and Lemma 25) we can find an element $e \in U$ with $p_e = \mathbb{P}(e \notin \mathbf{S} : \mathbf{S} \subseteq U) \geq \frac{1}{2m^2}$. Again using Lemma 25 we can perform step 2(c) and obtain a multiplicative $(1+\frac{\varepsilon}{m})$-approximation ρ_e to p_e.

Denote the set U at which the algorithm terminated by U' and the elements chosen at various stages of the algorithm by $e_1, e_2, ..., e_l$ with $l = m - |U'|$. The output of the algorithm is:

$$X \stackrel{\text{def}}{=} \rho_{e_1}\rho_{e_2}\cdot\dots\cdot p_{e_l}\mu(U').$$

While the exact value of the sum is

$$Z \stackrel{\text{def}}{=} p_{e_1}p_{e_2}\cdot\dots\cdot p_{e_l}\cdot \sum_{S\in\mathcal{C},S\subseteq U'} \mu(S).$$

Recall that for every $i = 1, 2, \dots, l$ with high probability it holds that:

$$\left(1+\frac{\varepsilon}{m}\right)^{-1} \leq \frac{p_{e_i}}{\rho_{e_i}} \leq \left(1+\frac{\varepsilon}{m}\right).$$

This, together with (7) implies that with high probability:

$$\left(1+\frac{\varepsilon}{m}\right)^{-l} \leq \frac{X}{Z} \leq \left(1+\frac{\varepsilon}{m}\right)^{l}\cdot\left(1+\frac{4}{m}\right),$$

which finally gives $(1+2\varepsilon)^{-1} \leq \frac{X}{Z} \leq (1+2\varepsilon)$ with high probability, as claimed. Note that the algorithm requires $\text{poly}(m, \frac{1}{\varepsilon})$ samples from the oracle in total. ◀

Sample-Based High-Dimensional Convexity Testing[*][†]

Xi Chen[1], **Adam Freilich**[2], **Rocco A. Servedio**[3], **and Timothy Sun**[4]

1 **Columbia University, New York, NY, USA**
`xichen@cs.columbia.edu`

2 **Columbia University, New York, NY, USA**
`freilich@cs.columbia.edu`

3 **Columbia University, New York, NY, USA**
`rocco@cs.columbia.edu`

4 **Columbia University, New York, NY, USA**
`tim@cs.columbia.edu`

Abstract

In the problem of *high-dimensional convexity testing*, there is an unknown set $S \subseteq \mathbb{R}^n$ which is promised to be either convex or ε-far from every convex body with respect to the standard multivariate normal distribution $\mathcal{N}(0,1)^n$. The job of a testing algorithm is then to distinguish between these two cases while making as few inspections of the set S as possible.

In this work we consider *sample-based* testing algorithms, in which the testing algorithm only has access to labeled samples $(\boldsymbol{x}, S(\boldsymbol{x}))$ where each $\boldsymbol{x}$ is independently drawn from $\mathcal{N}(0,1)^n$. We give nearly matching sample complexity upper and lower bounds for both one-sided and two-sided convexity testing algorithms in this framework. For constant ε, our results show that the sample complexity of one-sided convexity testing is $2^{\tilde{\Theta}(n)}$ samples, while for two-sided convexity testing it is $2^{\tilde{\Theta}(\sqrt{n})}$.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Property testing, convexity, sample-based testing

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.37

1 Introduction

Over the past few decades the field of property testing has developed into a fertile area with many different branches of active research. Several distinct lines of work have studied the testability of various kinds of *high-dimensional* objects, including probability distributions (see e.g. [9, 37, 4, 38, 2, 13, 1]), Boolean functions (see e.g. [17, 34, 15, 31, 28] and many other works), and various types of codes and algebraic objects (see e.g. [3, 23, 26, 14] and many other works). These efforts have collectively yielded significant insight into the abilities and limitations of efficient testing algorithms for such high-dimensional objects. A distinct line of work has focused on testing (mostly low-dimensional) *geometric properties*. Here too a considerable body of work has led to a good understanding of the testability of various low-dimensional geometric properties, see e.g. [19, 18, 36, 12, 11, 10].

This paper is about a topic which lies at the intersection of the two general strands (high-dimensional property testing and geometric property testing) mentioned above: we study the problem of *high-dimensional convexity testing*. Convexity is a fundamental property

[*] A full version of the paper is available at `https://arxiv.org/abs/1706.09362`.

[†] This work was supported by NSF grants CCF-1149257, CCF-1423100, CCF-1420349 and CCF-1563155.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 37; pp. 37:1–37:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

which is intensively studied in high-dimensional geometry (see e.g. [24, 8, 39] and many other references) and has been studied in the property testing of images (the two-dimensional case) [36, 12, 11, 10], but as we discuss in Section 1.2 below, very little is known about high-dimensional convexity testing.

We consider $\mathbb{R}^n$ endowed with the standard normal distribution $\mathcal{N}(0,1)^n$ as our underlying space, so the distance $\mathrm{dist}(S,C)$ between two subsets $S, C \subseteq \mathbb{R}^n$ is

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x}\leftarrow\mathcal{N}(0,1)^n}[\boldsymbol{x} \in S \triangle C],$$

where $S \triangle C$ denotes their symmetric difference. The standard normal distribution $\mathcal{N}(0,1)^n$ is arguably one of the most natural, and certainly one of the most studied, distributions on $\mathbb{R}^n$. Several previous works have studied property testing over $\mathbb{R}^n$ with respect to $\mathcal{N}(0,1)^n$, such as the work on testing halfspaces [31, 6] and the work on testing surface area [30, 33].

1.1 Our results

In this paper we focus on *sample-based* testing algorithms for convexity. Such an algorithm has access to independent draws $(\boldsymbol{x}, S(\boldsymbol{x})) \in \mathbb{R}^n \times \{0,1\}$, where $\boldsymbol{x}$ is drawn from $\mathcal{N}(0,1)^n$ and $S \subseteq \mathbb{R}^n$ is the unknown set being tested for convexity (so in particular the algorithm cannot select points to be queried) with $S(\boldsymbol{x}) = 1$ if $\boldsymbol{x} \in S$. We say such an algorithm is an *ε-tester for convexity* if it accepts S with probability at least $2/3$ when S is convex and rejects with probability at least $2/3$ when it is ε-far from convex, i.e., $\mathrm{dist}(S,C) \geq \varepsilon$ for all convex sets $C \subseteq \mathbb{R}^n$. The model of sample-based testing was originally introduced by Goldreich, Goldwasser, and Ron almost two decades ago [21], where it was referred to as "passive testing;" it has received significant attention over the years [27, 20, 6, 22], with an uptick in research activity in this model over just the past year or so [5, 16, 12, 11, 10].

We consider sample-based testers for convexity that are allowed both one-sided (i.e., the algorithm always accepts S when it is convex) and two-sided error. In each case, for constant $\varepsilon > 0$ we give nearly matching upper and lower bounds on sample complexity. Our results are as follows:

▶ **Theorem 1** (One-sided lower bound). *Any one-sided sample-based algorithm that is an ε-tester for convexity over $\mathcal{N}(0,1)^n$ for some $\varepsilon < 1/2$ must use $2^{\Omega(n)}$ samples.*

▶ **Theorem 2** (One-sided upper bound). *For any $\varepsilon > 0$, there is a one-sided sample-based ε-tester for convexity over $\mathcal{N}(0,1)^n$ which uses $(n/\varepsilon)^{O(n)}$ samples.*

▶ **Theorem 3** (Two-sided lower bound). *There exists a positive constant ε_0 such that any two-sided sample-based algorithm that is an ε-tester for convexity over $\mathcal{N}(0,1)^n$ for some $\varepsilon \leq \varepsilon_0$ must use $2^{\Omega(\sqrt{n})}$ samples.*

▶ **Theorem 4** (Two-sided upper bound). *For any $\varepsilon > 0$, there is a two-sided sample-based ε-tester for convexity over $\mathcal{N}(0,1)^n$ which uses $n^{O(\sqrt{n}/\varepsilon^2)}$ samples.*

These results are summarized in Table 1. We discuss the main ideas and techniques behind them in Section 1.3, and prove Theorem 3 in Section 3 and Theorem 1 in Section 4. We leave proofs of Theorems 2 and 4 to the full version due to space limitations.

1.2 Related work

Convexity testing. As discussed above, [36, 10, 11, 12] studied the testing of 2-dimensional convexity under the uniform distribution, either within a compact body such as $[0,1]^2$ [10, 11]

Table 1 Sample complexity bounds for sample-based convexity testing. In line four, $\varepsilon_0 > 0$ is some absolute constant.

Model	Sample complexity bound	Reference
One-sided	$2^{\Omega(n)}$ samples (for $\varepsilon < 1/2$)	Theorem 1
	$2^{O(n \log(n/\varepsilon))}$ samples	Theorem 2
Two-sided	$2^{\Omega(\sqrt{n})}$ samples (for $\varepsilon < \varepsilon_0$)	Theorem 3
	$2^{O(\sqrt{n} \log(n)/\varepsilon^2)}$ samples	Theorem 4

or over a discrete grid $[n]^2$ [36, 12]. The model of [10, 11] is more closely related to ours: [11] showed that $\Theta(\varepsilon^{-4/3})$ samples are necessary and sufficient for one-sided sample-based testers, while [10] gave a one-sided general tester (which can make adaptive queries to the unknown set) for 2-dimensional convexity with only $O(1/\varepsilon)$ queries.

The only prior work that we are aware of that deals with testing high-dimensional convexity is that of [35]. However, the model considered in [35] is different from ours in the following important aspects. First, the goal of an algorithm in their model is to determine whether an unknown $S \subseteq \mathbb{R}^n$ is not convex or is ε-close to convex in the following sense: the (Euclidean) volume of $S \triangle C$, for some convex C, is at most an ε-fraction of the volume of S. Second, in their model an algorithm both can make membership queries (to determine whether a given point x belongs to S), and can receive samples which are guaranteed to be drawn independently and uniformly at random from S. The main result of [35] is an algorithm which uses $(cn/\varepsilon)^n$ many random samples *drawn from* S, for some constant c, and $\text{poly}(n)/\varepsilon$ membership queries.

Sample-based testing. A wide range of papers have studied sample-based testing from several different perspectives, including the recent works [12, 11, 10] which study sample-based testing of convexity over two-dimensional domains. In earlier work on sample-based testing, [6] showed that the class of linear threshold functions can be tested to constant accuracy under $\mathcal{N}(0,1)^n$ with $\tilde{\Theta}(n^{1/2})$ samples drawn from $\mathcal{N}(0,1)^n$. (Note that a linear threshold function is a convex set of a very simple sort, as every convex set can be expressed as an intersection of (potentially infinitely many) linear threshold functions.) The work [6] in fact gave a characterization of the sample complexity of (two-sided) sample-based testing, in terms of a combinatorial/probabilistic quantity called the "passive testing dimension." This is a distribution-dependent quantity whose definition involves both the class being tested and the distribution from which samples are obtained; it is not *a priori* clear what the value of this quantity is for the class of convex subsets of $\mathbb{R}^n$ and the standard normal distribution $\mathcal{N}(0,1)^n$. Our upper and lower bounds (Theorems 4 and 3) may be interpreted as giving bounds on the passive testing dimension of the class of convex sets in $\mathbb{R}^n$ with respect to the $\mathcal{N}(0,1)^n$ distribution.

1.3 Our techniques

1.3.1 One-sided lower bound

Our one-sided lower bound has a simple proof using only elementary geometric and probabilistic arguments. It follows from the fact (see Lemma 17) that if $q = 2^{\Theta(n)}$ many points are drawn independently from $\mathcal{N}(0,1)^n$, then with probability $1-o(1)$ no one of the points lies in the convex hull of the $q-1$ others. This can easily be shown to imply that more than q samples are required (since given only q samples, with probability $1-o(1)$ there is a convex set consistent with any labeling and thus a one-sided algorithm cannot reject).

1.4 Two-sided lower bound

At a high-level, the proof of our two-sided lower bound uses the following standard approach. We first define two distributions $\mathcal{D}_{\mathsf{yes}}$ and $\mathcal{D}_{\mathsf{no}}$ over sets in $\mathbb{R}^n$ such that (i) $\mathcal{D}_{\mathsf{yes}}$ is a distribution over convex sets only, and (ii) $\mathcal{D}_{\mathsf{no}}$ is a distribution such that $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$ is ε_0-far from convex with probability at least $1-o(1)$ for some positive constant ε_0. We then show that every sample-based, q-query algorithm A with $q = 2^{0.01n}$ must have

$$\mathop{\mathbf{Pr}}_{\boldsymbol{S}\leftarrow\mathcal{D}_{\mathsf{yes}};\,\boldsymbol{x}}\big[A \text{ accepts } (\boldsymbol{x},\boldsymbol{S}(\boldsymbol{x}))\big] - \mathop{\mathbf{Pr}}_{\boldsymbol{S}\leftarrow\mathcal{D}_{\mathsf{no}};\,\boldsymbol{x}}\big[A \text{ accepts } (\boldsymbol{x},\boldsymbol{S}(\boldsymbol{x}))\big] \le o(1), \tag{1}$$

where $\boldsymbol{x}$ denotes a sequence of q points drawn from $\mathcal{N}(0,1)^n$ independently and $(\boldsymbol{x},\boldsymbol{S}(\boldsymbol{x}))$ denotes the q labeled samples from $\boldsymbol{S}$. Theorem 3 follows directly from (1).

To draw a set $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$, we sample a sequence of $N = 2^{\sqrt{n}}$ points $\mathbf{y}_1,\ldots,\mathbf{y}_N$ from the sphere $S^{n-1}(r)$ of radius r for some $r = \Theta(n^{1/4})$. Each $\mathbf{y}_i$ defines a halfspace $\boldsymbol{h}_i = \{x : x\cdot\mathbf{y}_i \le r^2\}$. $\boldsymbol{S}$ is then the intersection of all $\boldsymbol{h}_i$'s. (This is essentially a construction used by Nazarov [32] to exhibit a convex set that has large Gaussian surface area, and used by [29] to lower bound the sample complexity of learning convex sets under the Gaussian distribution.) The most challenging part of the two-sided lower bound proof is to show that, with q points $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_q \leftarrow \mathcal{N}(0,1)^n$, the q bits $\boldsymbol{S}(\boldsymbol{x}_1),\ldots,\boldsymbol{S}(\boldsymbol{x}_q)$ with $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$ are "almost" independent. More formally, the q bits $\boldsymbol{S}(\boldsymbol{x}_1),\ldots,\boldsymbol{S}(\boldsymbol{x}_q)$ with $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$ have $o(1)$-total variation distance from q independent bits with the ith bit drawn from the marginal distribution of $\boldsymbol{S}(\boldsymbol{x}_i)$ as $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$. On the other hand, it is relatively easy to define a distribution $\mathcal{D}_{\mathsf{no}}$ that satisfies (ii) and at the same time, $\boldsymbol{S}(\boldsymbol{x}_1),\ldots,\boldsymbol{S}(\boldsymbol{x}_q)$ when $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$ has $o(1)$-total variation distance from the same product distribution. (1) follows by combining the two parts.

1.4.1 Structural result

Our algorithms rely on a new structural result which we establish for convex sets in $\mathbb{R}^n$. Roughly speaking, this result gives an upper bound on the Gaussian volume of the "thickened surface" of any bounded convex subset of $\mathbb{R}^n$; it is inspired by, and builds on, the classic result of Ball [7] that upperbounds the Gaussian surface area of any convex subset of $\mathbb{R}^n$.

1.4.2 One-sided upper bound

Our one-sided testing algorithm employs a "gridding-based" approach to decompose the relevant portion of $\mathbb{R}^n$ (namely, those points which are not too far from the origin) into a collection of disjoint cubes. It draws samples and identifies a subset of these cubes as a proxy for the "thickened surface" of the target set; by the structural result sketched above, if the Gaussian volume of this thickened surface is too high, then the one-sided algorithm can safely reject (as the target set cannot be convex). Otherwise the algorithm does random sampling to probe for points which are inside the convex hull of positive examples it has received but are labeled negative (there should be no such points if the target set is indeed convex, so if such a point is identified, the one-sided algorithm can safely reject). If no such points are identified, then the algorithm accepts.

1.4.3 Two-sided upper bound

Finally, the main tool we use to obtain our two-sided testing algorithm is a *learning* algorithm for convex sets with respect to the normal distribution over $\mathbb{R}^n$. The main result of [29] is an (improper) algorithm which learns the class of all convex subsets of $\mathbb{R}^n$ to accuracy ε using $n^{O(\sqrt{n}/\varepsilon^2)}$ independent samples from $\mathcal{N}(0,1)^n$. Using the structural result mentioned above, we show that this can be converted into a *proper* algorithm for learning convex sets under $\mathcal{N}(0,1)^n$, with essentially no increase in the sample complexity. Given this proper learning algorithm, a two-sided algorithm for testing convexity follows from the well-known result of [21] which shows that proper learning for a class of functions implies (two-sided) testability.

2 Preliminaries and Notation

Notation. We use boldfaced letters such as $\boldsymbol{x}, \boldsymbol{f}, \mathbf{A}$, etc. to denote random variables (which may be real-valued, vector-valued, function-valued, set-valued, etc; the intended type will be clear from the context). We write "$\boldsymbol{x} \leftarrow \mathcal{D}$" to indicate that the random variable $\boldsymbol{x}$ is distributed according to probability distribution $\mathcal{D}$. Given $a, b, c \in \mathbb{R}$ we use $a = b \pm c$ to indicate that $b - c \le a \le b + c$.

Geometry. For $r > 0$, we write $S^{n-1}(r)$ to denote the origin-centered sphere of radius r in $\mathbb{R}^n$ and $\mathrm{Ball}(r)$ to denote the origin-centered ball of radius r in $\mathbb{R}^n$, i.e.,

$$S^{n-1}(r) = \big\{x \in \mathbb{R}^n : \|x\| = r\big\} \quad \text{and} \quad \mathrm{Ball}(r) = \big\{x \in \mathbb{R}^n : \|x\| \le r\big\},$$

where $\|x\|$ denotes the ℓ_2-norm $\|\cdot\|_2$ of x. We also write S^{n-1} for the unit sphere $S^{n-1}(1)$.

Recall that a set $C \subseteq \mathbb{R}^n$ is convex if $x, y \in C$ implies $\alpha x + (1-\alpha)y \in C$ for all $\alpha \in [0,1]$. We write $\mathcal{C}_{\text{convex}}$ to denote the class of all convex sets in $\mathbb{R}^n$. Recall that convex sets are Lebesgue measurable. Given a set $C \subseteq \mathbb{R}^n$ we use $\mathsf{Conv}(C)$ to denote the convex hull of C.

For sets $A, B \subseteq \mathbb{R}^n$, we write $A+B$ to denote the Minkowski sum $\{a+b : a \in A \text{ and } b \in B\}$. For a set $A \subseteq \mathbb{R}^n$ and $r > 0$ we write rA to denote the set $\{ra : a \in A\}$. Given a point a and $B \subseteq \mathbb{R}^n$, we use $a + B$ and $B - a$ to denote $\{a\} + B$ and $B + \{-a\}$ for convenience.

Probability. We use $\mathcal{N}(0,1)^n$ to denote the standard n-dimensional Gaussian distribution with zero mean and identity covariance matrix. We also recall that the probability density function for the one-dimensional Gaussian distribution is

$$\varphi(x) = \frac{1}{\sqrt{2\pi}} \cdot \exp(-x^2/2).$$

Sometimes we denote $\mathcal{N}(0,1)^n$ by $\mathcal{N}^n$ for convenience. The squared norm $\|\boldsymbol{x}\|^2$ of $\boldsymbol{x} \leftarrow \mathcal{N}(0,1)^n$ is distributed according to the chi-squared distribution χ^2_n with n degrees of freedom. The following tail bound for χ^2_n (see [25]) will be useful:

▶ **Lemma 5** (Tail bound for the chi-squared distribution)**.** *Let* $\mathbf{X} \leftarrow \chi^2_n$. *Then we have*

$$\mathbf{Pr}\big[|\mathbf{X} - n| \ge tn\big] \le e^{-(3/16)nt^2}, \quad \textit{for all } t \in [0, 1/2).$$

All target sets $S \subseteq \mathbb{R}^n$ to be tested for convexity are assumed to be Lebesgue measurable and we write $\mathrm{Vol}(S)$ to denote $\mathbf{Pr}_{\boldsymbol{x} \leftarrow \mathcal{N}^n}[\boldsymbol{x} \in S]$, the *Gaussian volume* of $S \subseteq \mathbb{R}^n$. Given two Lebesgue measurable subsets $S, C \subseteq \mathbb{R}^n$, we view $\mathrm{Vol}(S \triangle C)$ as the *distance* between S and C, where $S \triangle C$ is the symmetric difference of S and C. Given $S \subseteq \mathbb{R}^n$, we abuse the notation and use S to denote the indicator function of the set, so we may write "$S(x) = 1$" or "$x \in S$" to mean the same thing.

Sample-based property testing. Given a point $x \in \mathbb{R}^n$, we refer to $(x, S(x)) \in \mathbb{R}^n \times \{0,1\}$ as a *labeled sample* from a set $S \subseteq \mathbb{R}^n$. A *sample-based testing algorithm for convexity* is a randomized algorithm which is given as input an accuracy parameter $\varepsilon > 0$ and access to an oracle that, each time it is invoked, generates a labeled sample $(\boldsymbol{x}, S(\boldsymbol{x}))$ from the unknown (Lebesgue measurable) *target set* $S \subseteq \mathbb{R}^n$ with $\boldsymbol{x}$ drawn independently each time from $\mathcal{N}(0,1)^n$. When run with any Lebesgue measurable $S \subseteq \mathbb{R}^n$, such an algorithm must output "accept" with probability at least 2/3 (over the draws it gets from the oracle and its own internal randomness) if $S \in \mathcal{C}_{\text{convex}}$ and must output "reject" with probability at least 2/3 if S is ε*-far* from being convex, meaning that for every $C \in \mathcal{C}_{\text{convex}}$ it is the case that $\text{Vol}(S \triangle C) \geq \varepsilon$. (We also refer to an algorithm as an ε-tester for convexity if it works for a specific accuracy parameter ε.) Such a testing algorithm is said to be *one-sided* if whenever it is run on a convex set S it always outputs "accept;" equivalently, such an algorithm can only output "reject" if the labeled samples it receives are not consistent with any convex set. A testing algorithm which is not one-sided is said to be *two-sided.*

Throughout the rest of the paper we reserve the symbol S to denote the unknown target set (a measurable subset of $\mathbb{R}^n$) that is being tested for convexity.

3 Two-sided lower bound

We recall Theorem 3:

▶ **Theorem 3** (Two-sided lower bound). *There exists a positive constant ε_0 such that any two-sided sample-based algorithm that is an ε-tester for convexity over $\mathcal{N}(0,1)^n$ for some $\varepsilon \leq \varepsilon_0$ must use $2^{\Omega(\sqrt{n})}$ samples.*

Let $q = 2^{0.01\sqrt{n}}$ and let $\varepsilon_0 > 0$ be a constant to be specified later. To prove Theorem 3, we show that no sample-based, q-query (randomized) algorithm A can achieve the following:

> Let $S \subset \mathbb{R}^n$ be a target set that is Lebesgue measurable. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q$ be a sequence of q samples drawn from $\mathcal{N}(0,1)^n$. Upon receiving $((\boldsymbol{x}_i, S(\boldsymbol{x}_i)) : i \in [q])$, A accepts with probability at least 2/3 when S is convex and rejects with probability at least 2/3 when S is ε_0-far from convex.

Recall that a pair (x, b) with $x \in \mathbb{R}^n$ and $b \in \{0,1\}$ is a labeled sample; a sample-based algorithm A is a randomized map from a sequence of q labeled samples to {"accept","reject"}.

3.1 Proof Plan

Assume for contradiction that there is a q-query (randomized) algorithm A that accomplishes the task above. In Section 3.2 we define two probability distributions $\mathcal{D}_{\mathsf{yes}}$ and $\mathcal{D}_{\mathsf{no}}$ such that (1) $\mathcal{D}_{\mathsf{yes}}$ is a distribution over convex sets in $\mathbb{R}^n$ ($\mathcal{D}_{\mathsf{yes}}$ is a distribution over certain convex polytopes that are the intersection of many randomly drawn halfspaces), and (2) $\mathcal{D}_{\mathsf{no}}$ is a probability distribution over sets in $\mathbb{R}^n$ that are Lebesgue measurable ($\mathcal{D}_{\mathsf{no}}$ is actually supported over a finite number of measurable sets in $\mathbb{R}^n$) such that $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$ is ε_0-far from convex with probability at least $1 - o(1)$.

Given a sequence $x = (x_1, \ldots, x_q)$ of points, we abuse the notation and write

$$S(x) = (S(x_1), \ldots, S(x_q))$$

and use $(x, S(x))$ to denote the sequence of q labeled samples $(x_1, S(x_1)), \ldots, (x_q, S(x_q))$. It then follows from our assumption on A that

$$\Pr_{\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}};\, \boldsymbol{x} \leftarrow (\mathcal{N}^n)^q} \big[A \text{ accepts } (\boldsymbol{x}, \boldsymbol{S}(\boldsymbol{x}))\big] \geq 2/3 \qquad \text{and}$$

$$\Pr_{\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}};\, \boldsymbol{x} \leftarrow (\mathcal{N}^n)^q} \big[A \text{ accepts } (\boldsymbol{x}, \boldsymbol{S}(\boldsymbol{x}))\big] \leq 1/3 + o(1).$$

where we use $\boldsymbol{x} \leftarrow (\mathcal{N}^n)^q$ to denote a sequence of q points sampled independently from $\mathcal{N}^n$ and we usually skip the $\leftarrow (\mathcal{N}^n)^q$ part in the subscript when it is clear from the context. Since A is a mixture of deterministic algorithms, there exists a deterministic sample-based, q-query algorithm A' (equivalently, a deterministic map from sequences of q labeled samples to {"Yes", "No"}) with

$$\Pr_{\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}};\, \boldsymbol{x}} \big[A' \text{ accepts } (\boldsymbol{x}, \boldsymbol{S}(\boldsymbol{x}))\big] - \Pr_{\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}};\, \boldsymbol{x}} \big[A' \text{ accepts } (\boldsymbol{x}, \boldsymbol{S}(\boldsymbol{x}))\big] \geq 1/3 - o(1). \tag{2}$$

Let $\mathcal{E}_{\mathsf{yes}}$ (or $\mathcal{E}_{\mathsf{no}}$) be the distribution of $(\boldsymbol{x}, \boldsymbol{S}(\boldsymbol{x}))$, where $\boldsymbol{x} \leftarrow (\mathcal{N}^n)^q$ and $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$ (or $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$, respectively). Both of them are distributions over sequences of q labeled samples. Then the LHS of (2), for any deterministic sample-based, q-query algorithm A', is at most the total variation distance between $\mathcal{E}_{\mathsf{yes}}$ and $\mathcal{E}_{\mathsf{no}}$. We prove the following key lemma, which leads to a contradiction.

▶ **Lemma 6.** *The total variation distance between $\mathcal{E}_{\mathsf{yes}}$ and $\mathcal{E}_{\mathsf{no}}$ is $o(1)$.*

To prove Lemma 6, it will be convenient for us to introduce a third distribution $\mathcal{E}^*_{\mathsf{no}}$ over sequences of q labeled samples, where $(\boldsymbol{x}, \mathbf{b}) \leftarrow \mathcal{E}^*_{\mathsf{no}}$ is drawn by first sampling a sequence of q points $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q)$ from $\mathcal{N}^n$ independently and then for each $\boldsymbol{x}_i$, its label $\mathbf{b}_i$ is set to be 1 independently with a probability that depends only on $\|\boldsymbol{x}_i\|$ (see Section 3.2). Lemma 6 follows from the following two lemmas by the triangle inequality.

▶ **Lemma 7.** *The total variation distance between $\mathcal{E}_{\mathsf{no}}$ and $\mathcal{E}^*_{\mathsf{no}}$ is $o(1)$.*

▶ **Lemma 8.** *The total variation distance between $\mathcal{E}_{\mathsf{yes}}$ and $\mathcal{E}^*_{\mathsf{no}}$ is $o(1)$.*

The rest of the section is organized as follows. We define $\mathcal{D}_{\mathsf{yes}}$ and $\mathcal{D}_{\mathsf{no}}$ (which are used to define $\mathcal{E}_{\mathsf{yes}}$ and $\mathcal{E}_{\mathsf{no}}$) as well as $\mathcal{E}^*_{\mathsf{no}}$ in Section 3.2 and prove the necessary properties about $\mathcal{D}_{\mathsf{yes}}$ and $\mathcal{D}_{\mathsf{no}}$ as well as Lemma 7. We prove Lemma 8 in Sections 3.3 and Appendix A.

3.2 The Distributions

Let $r = \Theta(n^{1/4})$ be a parameter to be specified later, and let $N = 2^{\sqrt{n}}$. We start with the definition of $\mathcal{D}_{\mathsf{yes}}$. A random set $\boldsymbol{S} \subset \mathbb{R}^n$ is drawn from $\mathcal{D}_{\mathsf{yes}}$ using the following procedure:

1. We sample a sequence of N points $\mathbf{y}_1, \ldots, \mathbf{y}_N$ from $S^{n-1}(r)$ independently and uniformly at random. Each point $\mathbf{y}_i$ defines a halfspace

$$\boldsymbol{h}_i = \big\{x \in \mathbb{R}^n : x \cdot \mathbf{y}_i \leq r^2\big\}.$$

2. The set $\boldsymbol{S}$ is then the intersection of $\boldsymbol{h}_i$, $i \in [N]$ (this is always nonempty as indeed $\mathrm{Ball}(r)$ is contained in $\boldsymbol{S}$).

It is clear from the definition that $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$ is always a convex set.

Next we define $\mathcal{E}^*_{\mathsf{no}}$ (instead of $\mathcal{D}_{\mathsf{no}}$), a distribution over sequences of q labeled samples $(\boldsymbol{x}, \mathbf{b})$. To this end, we use $\mathcal{D}_{\mathsf{yes}}$ to define a function $\rho : \mathbb{R}_{\geq 0} \to [0, 1]$ as follows:

$$\rho(t) = \Pr_{\boldsymbol{S} \leftarrow \mathcal{D}_{\text{yes}}} \Big[(t, 0, \ldots, 0) \in \boldsymbol{S}\Big].$$

Due to the symmetry of $\mathcal{D}_{\mathsf{yes}}$ and $\mathcal{N}^n$, the value $\rho(t)$ is indeed the probability that a point $x \in \mathbb{R}^n$ at distance t from the origin lies in $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$. To draw a sequence of q labeled samples $(\boldsymbol{x}, \mathbf{b}) \leftarrow \mathcal{E}^*_{\mathsf{no}}$, we independently draw q random points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_q \leftarrow \mathcal{N}^n$ and then independently set $\mathbf{b}_i = 1$ with probability $\rho(\|\boldsymbol{x}_i\|)$ and $\mathbf{b}_i = 0$ with probability $1 - \rho(\|\boldsymbol{x}_i\|)$.

Given $\mathcal{D}_{\mathsf{yes}}$ and $\mathcal{E}^*_{\mathsf{no}}$, Lemma 8 shows that information-theoretically no sample-based algorithm can distinguish a sequence of q labeled samples $(\boldsymbol{x}, \mathbf{b})$ with $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$, $\boldsymbol{x} \leftarrow (\mathcal{N}^n)^q$, and $\mathbf{b} = \boldsymbol{S}(\boldsymbol{x})$ from a sequence of q labeled samples drawn from $\mathcal{E}^*_{\mathsf{no}}$. While the marginal distribution of each labeled sample is the same for the two cases, the former is generated in a correlated fashion using the underlying random convex $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$ while the latter is generated independently.

Finally we define the distribution $\mathcal{D}_{\mathsf{no}}$, prove Lemma 7, and show that a set drawn from $\mathcal{D}_{\mathsf{no}}$ is far from convex with high probability. To define $\mathcal{D}_{\mathsf{no}}$, we let $M \geq 2^{\sqrt{n}}$ be a large enough integer to be specified later. With M fixed, we use

$$0 = t_0 < t_1 < \cdots < t_{M-1} < t_M = 2\sqrt{n}$$

to denote a sequence of numbers such that the origin-centered ball $\mathrm{Ball}(2\sqrt{n})$ is partitioned into M *shells* $\mathrm{Ball}(t_i) \setminus \mathrm{Ball}(t_{i-1})$, $i \in [M]$, and all the M shells have the same probability mass under $\mathcal{N}^n$. By spherical coordinates, it means that the following integral takes the same value for all i:

$$\int_{t_{i-1}}^{t_i} \phi(x, 0, \ldots, 0) x^{n-1} dx, \tag{3}$$

where ϕ denotes the density function of $\mathcal{N}^n$. We show below that when M is large enough,

$$|\rho(x) - \rho(t_i)| \leq 2^{-\sqrt{n}}, \tag{4}$$

for any $i \in [M]$ and any $x \in [t_{i-1}, t_i]$. We will fix such an M and use it to define $\mathcal{D}_{\mathsf{no}}$. (Our results are not affected by the size of M as a function of n; we only need it to be finite.)

To show that (4) holds when M is large enough, we need the continuity of the function ρ, which follows directly from the explicit expression for ρ given later in (6).

▶ **Lemma 9.** *The function $\rho : \mathbb{R}_{\geq 0} \to [0, 1]$ is continuous.*

Since ρ is continuous, it is continuous over $[0, 2\sqrt{n}]$. Since $[0, 2\sqrt{n}]$ is compact, ρ is also uniformly continuous over $[0, 2\sqrt{n}]$. Also note that $\max_{i \in [M]}(t_i - t_{i-1})$ goes to 0 as M goes to $+\infty$. It follows that (4) holds when M is large enough.

With $M \geq 2^{\sqrt{n}}$ fixed, a random set $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$ is drawn as follows. Start with $\boldsymbol{S} = \emptyset$ and for each $i \in [M]$, add the ith shell $\mathrm{Ball}(t_i) \setminus \mathrm{Ball}(t_{i-1})$ to $\boldsymbol{S}$ independently with probability $\rho(t_i)$. Thus an outcome of $\boldsymbol{S}$ is a union of some of the shells and $\mathcal{D}_{\mathsf{no}}$ is supported over 2^M different sets.

Recall the definition of $\mathcal{E}_{\mathsf{yes}}$ and $\mathcal{E}_{\mathsf{no}}$ using $\mathcal{D}_{\mathsf{yes}}$ and $\mathcal{D}_{\mathsf{no}}$. We now prove Lemma 7.

Proof of Lemma 7. Let $x = (x_1, \ldots, x_q)$ be a sequence of q points in $\mathbb{R}^n$. We say x is *bad* if either (1) at least one point lies outside of $\mathrm{Ball}(2\sqrt{n})$ or (2) there are two points that lie in the same shell of $\mathcal{D}_{\mathsf{no}}$; we say x is *good* otherwise. We first claim that $\boldsymbol{x} \leftarrow (\mathcal{N}^n)^q$ is bad with probability $o(1)$. To see this, we have from Lemma 5 that event (1) occurs with probability $o(1)$, and from $M \geq 2^{\sqrt{n}}$ and $q = 2^{0.01\sqrt{n}}$ that event (2) occurs with probability $o(1)$. The claim follows from a union bound.

Given that $\boldsymbol{x} \leftarrow (\mathcal{N}^n)^q$ is good with probability $1 - o(1)$, it suffices to show that for any good q-tuple x, the total variation distance between (1) $\boldsymbol{S}(x)$ with $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$ and (2)

$\mathbf{b} = (\mathbf{b}_1, \ldots, \mathbf{b}_q)$ with each bit $\mathbf{b}_i$ being 1 with probability $\rho(\|x_i\|)$ independently, is $o(1)$. Let $\ell_i \in [M]$ be the index of the shell that x_i lies in. Since x is good (and thus, all points lie in different shells), $\boldsymbol{S}(x)$ has the ith bit being 1 independently with probability $\rho(t_{\ell_i})$; for the other distribution, the probability is $\rho(\|x_i\|)$. Using the subadditivity of total variation distance (i.e., the fact that the d_{TV} between two sequences of independent random variables is upper bounded by the sum of the d_{TV} between each pair) as well as (4), we have

$$d_{\mathrm{TV}}(\boldsymbol{S}(x), \mathbf{b}) \leq q \cdot 2^{-\sqrt{n}} = o(1).$$

This finishes the proof. ◀

The next lemma shows that $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$ is ε_0-far from convex with probability $1 - o(1)$, for some positive constant ε_0. In the proof of the lemma we fix both the constant ε_0 and our choice of $r = \Theta(n^{1/4})$. (We remind the reader that ρ and $\mathcal{D}_{\mathsf{no}}$ depend on the value of r.)

▶ **Lemma 10.** *There exist a real value $r = \Theta(n^{1/4})$ with $e^{r^2/2} \geq N/n$ and a positive constant ε_0 such that a set $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$ is ε_0-far from convex with probability at least $1 - o(1)$.*

Proof. We need the following claim but delay its proof to the end of the subsection:

▶ **Claim 11.** *There exist an $r = \Theta(n^{1/4})$ with $e^{r^2/2} \geq N/n$ and a constant $c \in (0, 1/2)$ such that $c < \rho(x) < 1 - c$ for all $x \in [\sqrt{n} - 10, \sqrt{n} + 10]$.*

Let $K \subset [M]$ denote the set of all integers k such that $[t_{k-1}, t_k] \subseteq [\sqrt{n} - 10, \sqrt{n} + 10]$ (note that K is a set of consecutive integers). Observe that (1) the total probability mass of all shells $k \in K$ is at least $\Omega(1)$ (by Lemma 5), and (2) the size $|K|$ is at least $\Omega(M)$ (which follows from (1) and the fact that all shells have the same probability mass).

Consider the following 1-dimensional scenario. We have $|K|$ intervals $[t_{k-1}, t_k]$ and draw $\boldsymbol{T}$ by including each interval independently with probability $\rho(t_k)$. We prove the following claim:

▶ **Claim 12.** *The random set $\boldsymbol{T}$ satisfies the following property with probability at least $1 - o(1)$: For any interval $I \subseteq \mathbb{R}_{\geq 0}$, either I contains $\Omega(M)$ intervals $[t_{k-1}, t_k]$ that are not included in $\boldsymbol{T}$, or $\overline{I}$ contains $\Omega(M)$ intervals $[t_{k-1}, t_k]$ included in $\boldsymbol{T}$.*

Proof. First note that it suffices to consider intervals $I \subseteq \cup_{k \in K} [t_{k-1}, t_k]$ and moreover, we may further assume that both endpoints of I come from endpoints of $[t_{k-1}, t_k]$, $k \in K$. (In other words, for a given outcome T of $\boldsymbol{T}$, if there exists an interval I that violates the condition, i.e., both I and $\overline{I}$ contain fewer than $\Omega(M)$ intervals, then there is such an interval I with both ends from end points of $[t_{k-1}, t_k]$). This assumption allows us to focus on $|K|^2 \leq M^2$ many possibilities for I (as we will see below, our argument applies a union bound over these K^2 possibilities).

Given a candidate such interval I, we consider two cases. If I contains $\Omega(M)$ intervals $[t_{k-1}, t_k]$, $k \in K$, then it follows from Claim 11 and a Chernoff bound that I contains at least $\Omega(M)$ intervals not included in $\boldsymbol{T}$ with probability $1 - 2^{-\Omega(M)}$. On the other hand, if $\overline{I}$ contains $\Omega(M)$ intervals, then the same argument shows that $\overline{I}$ contains $\Omega(M)$ interals included in $\boldsymbol{T}$ with probability $1 - 2^{-\Omega(M)}$. The claim follows from a union bound over all the $|K|^2$ possibilities for I. ◀

We return to the n-dimensional setting and consider the intersection of $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$ with a ray starting from the origin. Note that the intersection of the ray and any convex set is an interval on the ray. As a result, Claim 12 shows that with probability at least $1 - o(1)$ (over the draw of $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{no}}$), the intersection of any convex set with any ray either contains

$\Omega(M)$ intervals $[t_{k-1}, t_k]$ such that shell $k \in K$ is not included in $\boldsymbol{S}$, or misses $\Omega(M)$ intervals $[t_{k-1}, t_k]$ such that shell $k \in K$ is included in $\boldsymbol{S}$. Since by (1) above shells $k \in K$ together have $\Omega(1)$ probability mass under $\mathcal{N}^n$ and each shell contains the same probability mass, we have that with probability $1 - o(1)$, $\boldsymbol{S}$ is ε_0-far from any convex set for some constant $\varepsilon_0 > 0$. (A more formal argument can be given by performing integration using spherical coordinates and applying (3).) ◀

Proof of Claim 11. We start with the choice of r. Let

$$\alpha = \sqrt{n} - 10 \quad \text{and} \quad \beta = \sqrt{n} + 10.$$

Let $\text{cap}(t)$ denote the fractional surface area of the spherical cap $S^{n-1} \cap \{x : x_1 \geq t\}$, i.e.,

$$\text{cap}(t) = \mathop{\mathbf{Pr}}_{\boldsymbol{x} \leftarrow S^{n-1}} \big[\boldsymbol{x}_1 \geq t\big].$$

So cap is a continuous, strictly decreasing function over $[0, 1]$. Since $\text{cap}(0) = 1/2$ and $\text{cap}(1) = 0$, there is a unique $r \in (0, \alpha)$ such that $\text{cap}(r/\alpha) = 1/N = 2^{-\sqrt{n}}$. Below we show that $r = \Theta(n^{1/4})$ and fix it in the rest of the proof. First recall the following explicit expression (see e.g. [29]):

$$\text{cap}(t) = a_n \int_t^1 \left(\sqrt{1 - z^2}\right)^{n-3} dz,$$

where $a_n = \Theta(n^{1/2})$ is a parameter that only depends on n. We also recall the following inequalities from [29] about $\text{cap}(t)$:

$$\text{cap}(t) \leq e^{-nt^2/2}, \quad \text{for all } t \in [0, 1]; \quad \text{cap}(t) \geq \Omega\left(t \cdot e^{-nt^2/2}\right), \quad \text{for } t = O(1/n^{1/4}). \tag{5}$$

By our choice of α and the monotonicity of the cap function, we have $r = \Theta(n^{1/4})$ and

$$\begin{aligned} 1/N = \text{cap}(r/\alpha) &\geq \Omega(1/n^{1/4}) \cdot e^{-n(r/\alpha)^2/2} \\ &\geq \Omega(1/n^{1/4}) \cdot e^{-(r^2/2)(1+O(1/\sqrt{n}))} = \Omega(1/n^{1/4}) \cdot e^{-r^2/2} \end{aligned}$$

(using $r = \Theta(n^{1/4})$ for the last inequality), and thus, we have $e^{r^2/2} \geq N/n$.

Next, using the function cap we have the following expression for ρ:

$$\rho(x) = \left(1 - \text{cap}\left(\frac{r}{x}\right)\right)^N. \tag{6}$$

As a side note, ρ is continuous and thus, Lemma 9 follows. Since cap is strictly decreasing, we have that ρ is strictly decreasing as well. To finish the proof it suffices to show that there is a constant $c \in (0, 1/2)$ such that $\rho(\alpha) < 1 - c$ and $\rho(\beta) \geq c$.

$$\rho(\alpha) = (1 - 1/N)^N \approx e^{-1}$$

by our choice of r. In the rest of the proof we show that

$$\text{cap}\left(\frac{r}{\beta}\right) \leq a \cdot \text{cap}\left(\frac{r}{\alpha}\right) = \frac{a}{N}, \tag{7}$$

for some positive constant a. It follows immediately that

$$\rho(\beta) = \left(1 - \text{cap}\left(\frac{r}{\beta}\right)\right)^N \geq \left(1 - \frac{a}{N}\right)^N \geq \left(e^{-2a/N}\right)^N = e^{-2a},$$

using $1 - x \geq e^{-2x}$ for $0 \leq x \ll 1$, and this finishes the proof of the claim.

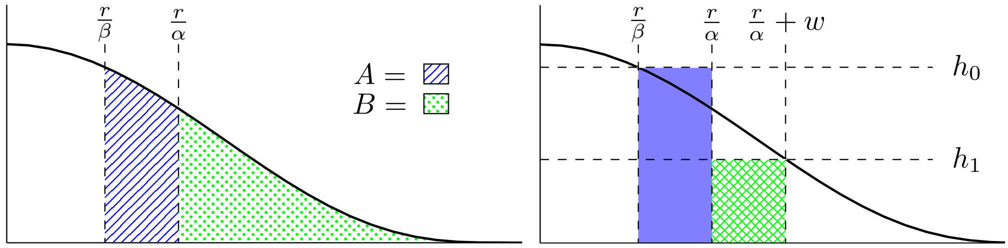


Figure 1 A plot of the integrand $(\sqrt{1-z^2})^{(n-3)}$. Area A is $\mathrm{cap}(r/\beta) - \mathrm{cap}(r/\alpha)$ and area B is $\mathrm{cap}(r/\alpha)$. The rectangles on the right are an upper bound of A and a lower bound of B.

Finally we prove (7). Let

$$w = \frac{r}{\alpha} - \frac{r}{\beta} = \Theta\left(\frac{1}{n^{3/4}}\right)$$

since $r = \Theta(n^{1/4})$. Below we show that

$$\int_{r/\beta}^{r/\alpha} \left(\sqrt{1-z^2}\right)^{n-3} dz \le a' \cdot \int_{r/\alpha}^{r/\alpha+w} \left(\sqrt{1-z^2}\right)^{n-3} dz, \tag{8}$$

for some positive constant a'. It follows that

$$\mathrm{cap}\left(\frac{r}{\beta}\right) - \mathrm{cap}\left(\frac{r}{\alpha}\right) \le a' \cdot \mathrm{cap}\left(\frac{r}{\alpha}\right)$$

and implies (7) by setting $a = a' + 1$. For (8), note that the ratio of the $[r/\beta, r/\alpha]$-integration over the $[r/\alpha, r/\alpha{+}w]$-integration is at most

$$\left(\frac{\sqrt{1-(r/\beta)^2}}{\sqrt{1-(r/\beta+2w)^2}}\right)^{n-3}$$

as the length of the two intervals are the same and the function $(\sqrt{1-z^2})^{n-3}$ is strictly decreasing. Figure 1 illustrates this calculation.

Let $\tau = r/\beta = \Theta(1/n^{1/4})$. We can rewrite the above as

$$\left(\frac{1-\tau^2}{1-(\tau+2w)^2}\right)^{(n-3)/2} = \left(1 + \frac{4\tau w + 4w^2}{1-(\tau+2w)^2}\right)^{(n-3)/2} = \left(1 + O\left(\frac{1}{n}\right)\right)^{(n-3)/2} = O(1).$$

This finishes the proof of the claim. ◀

3.3 Distributions $\mathcal{E}_{\mathsf{yes}}$ and $\mathcal{E}^*_{\mathsf{no}}$ are close

In the rest of the section we show that the total variation distance between $\mathcal{E}_{\mathsf{yes}}$ and $\mathcal{E}^*_{\mathsf{no}}$ is $o(1)$ and thus prove Lemma 8. Let $z = (z_1, \ldots, z_q)$ be a sequence of q points in $\mathbb{R}^n$. We use $\mathcal{E}_{\mathsf{yes}}(z)$ to denote the distribution of labeled samples from $\mathcal{E}_{\mathsf{yes}}$, conditioning on the samples being z, i.e., $(z, \boldsymbol{S}(z))$ with $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$. We let $\mathcal{E}^*_{\mathsf{no}}(z)$ denote the distribution of labeled samples from $\mathcal{E}^*_{\mathsf{no}}$, conditioning on the samples being z, i.e., $(z, \mathbf{b})$ where each $\mathbf{b}_i$ is 1 independently with probability $\rho(\|z_i\|)$. Then

$$d_{\mathrm{TV}}(\mathcal{E}_{\mathsf{yes}}, \mathcal{E}^*_{\mathsf{no}}) = \mathbf{E}_{\mathbf{z}\leftarrow(\mathcal{N}^n)^q}\left[d_{\mathrm{TV}}(\mathcal{E}_{\mathsf{yes}}(\mathbf{z}), \mathcal{E}^*_{\mathsf{no}}(\mathbf{z}))\right]. \tag{9}$$

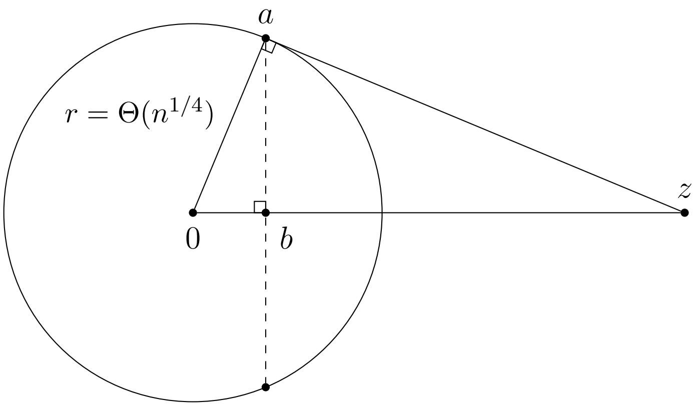


Figure 2 The fractional surface area of $\mathrm{cover}(z)$, $\mathrm{fsa}(\mathrm{cover}(z))$, is the fraction of $S^{n-1}(r)$ to the right of the dashed line. By similarity of triangles $0az$ and $0ba$, scaling down to the unit sphere, we get (10).

We split the proof of Lemma 8 into two steps. We first introduce the notion of *typical* sequences z of q points and show in this subsection that with probability $1 - o(1)$, $\mathbf{z} \leftarrow (\mathcal{N}^n)^q$ is typical. In the next subsection we show that $d_{\mathrm{TV}}(\mathcal{E}_{\mathsf{yes}}(z), \mathcal{E}^*_{\mathsf{no}}(z))$ is $o(1)$ when z is typical. It follows from (9) that $d_{\mathrm{TV}}(\mathcal{E}_{\mathsf{yes}}, \mathcal{E}^*_{\mathsf{no}}) = o(1)$. We start with the definition of typical sequences.

Given a point $z \in \mathbb{R}^n$, we are interested in the *fraction* of points y (in terms of the area) in $S^{n-1}(r)$ such that $z \cdot y > r^2$. This is because if any such point y is sampled in the construction of $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$, then $z \notin \boldsymbol{S}$. This is illustrated in Figure 2. We refer to the set of such points y as the (*spherical*) *cap covered by* z and we write $\mathrm{cover}(z)$ to denote it. (Note that $\mathrm{cover}(z) = \emptyset$ if $\|z\| \le r$.)

Given a subset H of $S^{n-1}(r)$ (such as $\mathrm{cover}(z)$), we use $\mathrm{fsa}(H)$ to denote the fractional surface area of H with respect to $S^{n-1}(r)$. Using Figure 2 and elementary geometry, we have the following connection between the fractional surface area of $\mathrm{cover}(z)$ and the cap function (for S^{n-1}):

$$\mathrm{fsa}\big(\mathrm{cover}(z)\big) = \mathrm{cap}\big(r/\|z\|\big). \tag{10}$$

We are now ready to define typical sequences.

▶ **Definition 13.** We say a sequence $z = (z_1, \ldots, z_q)$ of q points in $\mathbb{R}^n$ is *typical* if

1. For every point z_i, we have

$$\mathrm{fsa}\big(\mathrm{cover}(z_i)\big) \in \Big[e^{-0.51\,r^2}, e^{-0.49\,r^2}\Big]. \tag{11}$$

2. For every $i \neq j$, we have

$$\mathrm{fsa}\big(\mathrm{cover}(z_i) \cap \mathrm{cover}(z_j)\big) \le e^{-0.96\,r^2}.$$

The first condition of typicality essentially says that every z_i is not too close to and not too far away from the origin (so that we have a relatively tight bound on the fractional surface area of the cap covered by z_i). The second condition says that the caps covered by two points z_i and z_j have very little intersection. We prove the following lemma:

▶ **Lemma 14.** $\mathbf{z} \leftarrow (\mathcal{N}^n)^q$ *is typical with probability at least* $1 - o(1)$.

Proof. We show that $\mathbf{z}$ satisfies each of the two conditions with probability $1 - o(1)$. The lemma then follows from a union bound.

For the first condition, we let $c^* = 0.001$ be a sufficiently small constant. We have from Lemma 5 and a union bound that every $\mathbf{z}_i$ satisfies $(1 - c^*)\sqrt{n} \le \|\mathbf{z}_i\| \le (1 + c^*)\sqrt{n}$ with probability $1 - o(1)$. When this happens, we have (11) for every $\mathbf{z}_i$ using (5) and the upper bound of $\mathrm{cap}(t) \le e^{-nt^2/2}$.

For the second condition, we note that the argument used in the first part implies that

$$\mathbf{E}_{\boldsymbol{z}_i \leftarrow \mathcal{N}^n}\Big[\mathrm{fsa}\big(\mathrm{cover}(\boldsymbol{z}_i)\big)\Big] \le e^{-0.49\,r^2}.$$

Let x_0 be a fixed point in $S^{n-1}(r)$. Viewing the fsa as the following probability:

$$\mathrm{fsa}\big(\mathrm{cover}(z_i)\big) = \mathop{\mathbf{Pr}}_{\boldsymbol{x} \leftarrow S^{n-1}(r)}\big[\boldsymbol{x} \in \mathrm{cover}(z_i)\big],$$

we have

$$\begin{aligned} e^{-0.49\,r^2} &\ge \mathbf{E}_{\boldsymbol{z}_i \leftarrow \mathcal{N}^n}\Big[\mathrm{fsa}\big(\mathrm{cover}(\boldsymbol{z}_i)\big)\Big] \qquad (12)\\ &= \mathbf{E}_{\boldsymbol{z}_i}\Big[\mathop{\mathbf{Pr}}_{\boldsymbol{x} \leftarrow S^{n-1}(r)}\big[\boldsymbol{x} \in \mathrm{cover}(\mathbf{z}_i)\big]\Big]\\ &= \mathop{\mathbf{Pr}}_{\boldsymbol{x},\mathbf{z}_i}\big[\boldsymbol{x} \in \mathrm{cover}(\mathbf{z}_i)\big] = \mathop{\mathbf{Pr}}_{\mathbf{z}_i}\big[x_0 \in \mathrm{cover}(\mathbf{z}_i)\big], \end{aligned}$$

where the last equation follows by sampling $\boldsymbol{x}$ first and spherical and Gaussian symmetry.

Similarly we can express the fractional surface area of $\mathrm{cover}(z_i) \cap \mathrm{cover}(z_j)$ as

$$\mathrm{fsa}\big(\mathrm{cover}(z_i) \cap \mathrm{cover}(z_j)\big) = \mathop{\mathbf{Pr}}_{\boldsymbol{x} \leftarrow S^{n-1}(r)}\big[\boldsymbol{x} \in \mathrm{cover}(z_i) \text{ and } \boldsymbol{x} \in \mathrm{cover}(z_j)\big].$$

We consider the expectation over $\mathbf{z}_i$ and $\mathbf{z}_j$ drawn independently from $\mathcal{N}^n$:

$$\begin{aligned} &\mathbf{E}_{\mathbf{z}_i,\mathbf{z}_j}\Big[\mathrm{fsa}\big(\mathrm{cover}(\mathbf{z}_i) \cap \mathrm{cover}(\mathbf{z}_j)\big)\Big]\\ &= \mathbf{E}_{\mathbf{z}_i,\mathbf{z}_j}\Big[\mathop{\mathbf{Pr}}_{\boldsymbol{x} \leftarrow S^{n-1}(r)}\big[\boldsymbol{x} \in \mathrm{cover}(\mathbf{z}_i) \text{ and } \boldsymbol{x} \in \mathrm{cover}(\mathbf{z}_j)\big]\Big]\\ &= \mathop{\mathbf{Pr}}_{\boldsymbol{x},\mathbf{z}_i,\mathbf{z}_j}\big[\boldsymbol{x} \in \mathrm{cover}(\mathbf{z}_i) \text{ and } \boldsymbol{x} \in \mathrm{cover}(\mathbf{z}_j)\big]\\ &= \mathop{\mathbf{Pr}}_{\mathbf{z}_i}\big[x_0 \in \mathrm{cover}(\mathbf{z}_i)\big] \cdot \mathop{\mathbf{Pr}}_{\mathbf{z}_j}\big[x_0 \in \mathrm{cover}(\mathbf{z}_j)\big], \end{aligned}$$

where the last equation follows by sampling $\boldsymbol{x}$ first, independence of $\mathbf{z}_i, \mathbf{z}_j$, and symmetry.

By (12), the expectation of $\mathrm{fsa}(\mathrm{cover}(\mathbf{z}_i) \cap \mathrm{cover}(\mathbf{z}_j))$ is at most $e^{-0.98\,r^2}$, and hence by Markov's inequality, the probability of it being at least $e^{-0.96\,r^2}$ is at most $e^{-0.02\,r^2}$. Using $e^{r^2} \ge (N/n)^2$ and a union bound, the probability of one of the pairs having the fsa at least $e^{-0.96\,r^2}$ is at most

$$q^2 \cdot e^{-0.02r^2} \le 2^{0.02\sqrt{n}} \cdot (n/N)^{0.04} = o(1),$$

since $q = 2^{0.01\sqrt{n}}$ and $N = 2^{\sqrt{n}}$. This finishes the proof of the lemma. ◀

We prove the following lemma in Appendix A to finish the proof of Lemma 8.

▶ **Lemma 15.** *For every typical sequence z of q points, we have*

$$d_{TV}\big(\mathcal{E}_{\textsf{yes}}(z), \mathcal{E}^*_{\textsf{no}}(z)\big) = o(1).$$

4 One-sided lower bound

We recall Theorem 1:

▶ **Theorem 1** (One-sided lower bound). *Any one-sided sample-based algorithm that is an ε-tester for convexity over $\mathcal{N}(0,1)^n$ for some $\varepsilon < 1/2$ must use $2^{\Omega(n)}$ samples.*

We say a finite set $\{x^1, \ldots, x^M\} \subset \mathbb{R}^n$ is *shattered* by $\mathcal{C}_{\text{convex}}$ if for every $(b_1, \ldots, b_M) \in \{0,1\}^M$ there is a convex set $C \in \mathcal{C}_{\text{convex}}$ such that $C(x^i) = b_i$ for all $i \in [M]$. Theorem 1 follows from the following lemma:

▶ **Lemma 16.** *There is an absolute constant $c > 0$ such that for $M = 2^{cn}$, it holds that*

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x}^i \leftarrow \mathcal{N}(0,1)^n}\left[\{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^M\} \text{ is shattered by } \mathcal{C}_{\text{convex}}\right] \geq 1 - o(1).$$

Proof of Theorem 1 using Lemma 16. Suppose that A were a one-sided sample-based algorithm for ε-testing $\mathcal{C}_{\text{convex}}$ using at most M samples. Fix a set S that is ε-far from $\mathcal{C}_{\text{convex}}$ to be the unknown target subset of $\mathbb{R}^n$ that is being tested.[1] Since S is ε-far from convex, it must be the case that

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x}^i \leftarrow \mathcal{N}(0,1)^n}\left[A \text{ rejects } (\boldsymbol{x}^1, S(\boldsymbol{x}^1)), \ldots, (\boldsymbol{x}^M, S(\boldsymbol{x}^M))\right] \geq 2/3. \tag{13}$$

But Lemma 16 together with the one-sidedness of A imply that

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x}^i \leftarrow \mathcal{N}(0,1)^n}\left[\text{for any } (b^1, \ldots, b^M) \in \{0,1\}^M,\ A \text{ rejects } (\boldsymbol{x}^1, b^1), \ldots, (\boldsymbol{x}^M, b^M)\right] \leq o(1),$$

since A can only reject if the labeled samples are not consistent with any convex set, which implies that A cannot reject when $\{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^M\}$ is shattered by $\mathcal{C}_{\text{convex}}$. This contradicts with (13) and finishes the proof of the lemma. ◀

In the next subsection we prove Lemma 16 for $c = 1/500$.

4.1 Proof of Lemma 16

Let $M = 2^{cn}$ with $c = 1/500$. We prove the following lemma:

▶ **Lemma 17.** *For $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^M$ drawn independently from $\mathcal{N}(0,1)^n$, with probability $1 - o(1)$ it is the case that for all $i \in [M]$, no $\boldsymbol{x}^i$ lies in $\mathsf{Conv}(\{\boldsymbol{x}^j : j \in [M] \setminus i\})$.*

If $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^M$ are such that no $\boldsymbol{x}^i$ lies in $\mathsf{Conv}(\{\boldsymbol{x}^j : j \in [M] \setminus i\})$, then given any tuple $(b^1, \ldots, b^M)$, by taking $C = \mathsf{Conv}(\{\boldsymbol{x}^i : b^i = 1\})$ we see that there is a convex set C such that $C(\boldsymbol{x}^i) = b^i$ for all $i \in [M]$. Thus to establish Lemma 16 it suffices to prove Lemma 17.

To prove Lemma 17, it suffices to show that for each fixed $j \in [M]$ we have

$$\mathop{\mathbf{Pr}}_{\boldsymbol{x}^i \leftarrow \mathcal{N}(0,1)^n}\left[\boldsymbol{x}^j \in \mathsf{Conv}(\{\boldsymbol{x}^k : k \in [M] \setminus \{j\}\})\right] \leq M^{-2} \tag{14}$$

[1] An example of such a subset S is as follows (we define it as a function $S : \mathbb{R}^n \to \{0,1\}$): Given an odd integer $N > (1/2 - \varepsilon)^{-1} - 1$, let $-\infty = \tau_0 < \tau_1 < \cdots < \tau_N < \tau_{N+1} = +\infty$ be values such that $\mathbf{Pr}_{\boldsymbol{z} \leftarrow \mathcal{N}(0,1)}[\boldsymbol{z} \leq \tau_i] = i/(N+1)$, and let $S : \mathbb{R}^n \to \{0,1\}$ be the function defined by $S(x_1, \ldots, x_n) = \mathbf{1}[i \text{ is even}]$, where $i \in \{0, \ldots, N\}$ is the unique value such that $\tau_i \leq x_1 < \tau_{i+1}$. Fix any $z = (z_2, \ldots, z_n) \in \mathbb{R}^{n-1}$ and we let $S_z : \mathbb{R} \to \{0,1\}$ be the function defined as $S_z(x_1) = S(x_1, z_2, \ldots, z_n)$. An easy argument gives that S_z is $(1/2 - 1/(N+1))$-far (and hence ε-far) from every convex subset of $\mathbb{R}$, and it follows by averaging (using the fact that the restriction of any convex subset of $\mathbb{R}^n$ to a line is a convex subset of $\mathbb{R}$) that S is ε-far from $\mathcal{C}_{\text{convex}}$.

since given this a union bound implies that

$$\Pr_{\boldsymbol{x}^i \leftarrow \mathcal{N}(0,1)^n}\left[\text{for some } j \in [M],\ \boldsymbol{x}^j \text{ lies in } \mathsf{Conv}(\{\boldsymbol{x}^k : k \in [M] \setminus \{j\}\})\right] \le M^{-1} = o(1).$$

By symmetry, to establish (14) it suffices to show that

$$\Pr_{\boldsymbol{x}^i \leftarrow \mathcal{N}(0,1)^n}\left[\boldsymbol{x}^M \in \mathsf{Conv}(\{\boldsymbol{x}^1, \dots, \boldsymbol{x}^{M-1}\})\right] \le M^{-2}. \tag{15}$$

In turn (15) follows from the following inequalities (v is a fixed unit vector in the second)

$$\Pr_{\boldsymbol{x} \leftarrow \mathcal{N}(0,1)^n}\left[\|\boldsymbol{x}\| \le \sqrt{n}/10\right] < \frac{1}{2}M^{-2} \quad \text{and} \quad \Pr_{\boldsymbol{x} \leftarrow \mathcal{N}(0,1)^n}\left[\boldsymbol{x} \cdot v \ge \sqrt{n}/10\right] < \frac{1}{2}M^{-3}. \tag{16}$$

The first inequality follows from Lemma 5 using $c = 1/500$. For the second, by the spherical symmetry of $\mathcal{N}(0,1)^n$ we may take $v = (1, 0, \dots, 0)$. Recall the standard Gaussian tail bound

$$\Pr_{\boldsymbol{z} \leftarrow \mathcal{N}(0,1)}\left[\boldsymbol{z} \ge t\right] \le e^{-t^2/2}$$

for $t \ge 0$. This gives us that

$$\Pr_{\boldsymbol{x} \leftarrow \mathcal{N}(0,1)^n}\left[\boldsymbol{x} \cdot v \ge \sqrt{n}/10\right] \le e^{-n/200} < \frac{1}{2}M^{-3},$$

again using that $M = 2^{cn}$ and $c = 1/500$.

Finally, to see that (15) follows from (16), we observe first that by the first inequality we may assume that $\|\boldsymbol{x}^M\| > \sqrt{n}/10$ (at the cost of failure probability at most $M^{-2}/2$ towards (15)); fix any such outcome x^M of $\boldsymbol{x}^M$. By a union bound over $\boldsymbol{x}^1, \dots, \boldsymbol{x}^{M-1}$ and the second inequality, we have

$$\Pr_{\boldsymbol{x}^i \leftarrow \mathcal{N}(0,1)^n}\left[\text{any } i \in [M-1] \text{ has } \boldsymbol{x}^i \cdot \frac{x^M}{\|x^M\|} \ge \sqrt{n}/10\right] < \frac{1}{2}M^{-2}.$$

But if every $\boldsymbol{x}^i$ has $\boldsymbol{x}^i \cdot (x^M/\|x^M\|) < \sqrt{n}/10 < \|x^M\|$, then $x^M \notin \mathsf{Conv}(\{\boldsymbol{x}^1, \dots, \boldsymbol{x}^{M-1}\})$.

References

1 Jayadev Acharya, Constantinos Daskalakis, and Gautam Kamath. Optimal testing for properties of distributions. In *Advances in Neural Information Processing Systems 28 (NIPS)*, pages 3591–3599, 2015.

2 Michal Adamaszek, Artur Czumaj, and Christian Sohler. Testing monotone continuous distributions on high-dimensional real cubes. In *SODA*, pages 56–65, 2010.

3 N. Alon, T. Kaufman, M. Krivelevich, S. Litsyn, and D. Ron. Testing Reed-Muller Codes. *IEEE Transactions on Information Theory*, 51(11):4032–4039, 2005.

4 Noga Alon, Alexandr Andoni, Tali Kaufman, Kevin Matulef, Ronitt Rubinfeld, and Ning Xie. Testing k-wise and almost k-wise independence. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 496–505, 2007.

5 Noga Alon, Rani Hod, and Amit Weinstein. On active and passive testing. *Combinatorics, Probability & Computing*, 25(1):1–20, 2016.

6 Maria-Florina Balcan, Eric Blais, Avrim Blum, and Liu Yang. Active property testing. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 21–30, 2012.

7 K. Ball. The Reverse Isoperimetric Problem for Gaussian Measure. *Discrete and Computational Geometry*, 10:411–420, 1993.

8 Keith Ball. An elementary introduction to modern convex geometry. In *Flavors of Geometry*, pages 1–58. MSRI Publications, 1997.

9 Tugkan Batu, Ravi Kumar, and Ronitt Rubinfeld. Sublinear algorithms for testing monotone and unimodal distributions. In *Proceedings of the 36th Symposium on Theory of Computing*, pages 381–390, 2004.

10 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. The power and limitations of uniform samples in testing properties of figures. In *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2016, December 13-15, 2016, Chennai, India*, pages 45:1–45:14, 2016.

11 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Testing convexity of figures under the uniform distribution. In *32nd International Symposium on Computational Geometry, SoCG 2016, June 14-18, 2016, Boston, MA, USA*, pages 17:1–17:15, 2016.

12 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Tolerant testers of image properties. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 90:1–90:14, 2016.

13 Arnab Bhattacharyya, Eldar Fischer, Ronitt Rubinfeld, and Paul Valiant. Testing monotonicity of distributions over general partial orders. In *ICS*, pages 239–252, 2011.

14 Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of reed-muller codes. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010*, pages 488–497, 2010.

15 Eric Blais. Testing juntas nearly optimally. In *Proc. 41st Annual ACM Symposium on Theory of Computing (STOC)*, pages 151–158, 2009. doi:10.1145/1536414.1536437.

16 Eric Blais and Yuichi Yoshida. A characterization of constant-sample testable properties. *CoRR*, abs/1612.06016, 2016. URL: http://arxiv.org/abs/1612.06016.

17 M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47:549–595, 1993. Earlier version in STOC'90.

18 Artur Czumaj and Christian Sohler. Property testing with geometric queries. In *Algorithms – ESA 2001, 9th Annual European Symposium*, pages 266–277, 2001.

19 Artur Czumaj, Christian Sohler, and Martin Ziegler. Property testing in computational geometry. In *Algorithms – ESA 2000, 8th Annual European Symposium*, pages 155–166, 2000.

20 O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samordinsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

21 O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45:653–750, 1998.

22 Oded Goldreich and Dana Ron. On sample-based testers. *TOCT*, 8(2):7:1–7:54, 2016.

23 Oded Goldreich and Madhu Sudan. Locally testable codes and pcps of almost-linear length. *J. ACM*, 53(4):558–655, 2006.

24 P.M. Gruber and J.M. Wills, editors. *Handbook of convex geometry, Volume A*. Elsevier, New York, 1993.

25 Iain M. Johnstone. Chi-square oracle inequalities. In *State of the art in probability and statistics*, pages 399–418. Institute of Mathematical Statistics, 2001.

26 Tali Kaufman and Madhu Sudan. Algebraic property testing: the role of invariance. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 403–412, 2008.

27 M. Kearns and D. Ron. Testing problems with sub-learning sample complexity. *Journal of Computer and System Sciences*, 61:428–456, 2000.

28 Subhash Khot, Dor Minzer, and Muli Safra. On monotonicity testing and boolean isoperimetric type theorems. To appear in FOCS, 2015.

29 A. Klivans, R. O'Donnell, and R. Servedio. Agnostically learning convex sets via perimeter. manuscript, 2007.

30 Pravesh Kothari, Amir Nayyeri, Ryan O'Donnell, and Chenggang Wu. Testing surface area. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1204–1214, 2014.

31 K. Matulef, R. O'Donnell, R. Rubinfeld, and R. Servedio. Testing halfspaces. *SIAM J. on Comput.*, 39(5):2004–2047, 2010.

32 F. Nazarov. On the maximal perimeter of a convex set in $\mathbb{R}^n$ with respect to a Gaussian measure. In *Geometric aspects of functional analysis (2001-2002)*, pages 169–187. Lecture Notes in Math., Vol. 1807, Springer, 2003.

33 Joe Neeman. Testing surface area with arbitrary accuracy. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC'14, pages 393–397, 2014.

34 M. Parnas, D. Ron, and A. Samorodnitsky. Testing Basic Boolean Formulae. *SIAM J. Disc. Math.*, 16:20–46, 2002. URL: `citeseer.ifi.unizh.ch/parnas02testing.html`.

35 Luis Rademacher and Santosh Vempala. Testing geometric convexity. In *FSTTCS 2004: Foundations of Software Technology and Theoretical Computer Science: 24th International Conference, Chennai, India, December 16-18, 2004. Proceedings*, pages 469–480, 2005.

36 Sofya Raskhodnikova. Approximate testing of visual properties. In *Proceedings of RANDOM*, pages 370–381, 2003.

37 R. Rubinfeld and R. Servedio. Testing monotone high-dimensional distributions. In *Proc. 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 147–156, 2005.

38 Ronitt Rubinfeld and Ning Xie. Testing non-uniform k-wise independent distributions over product spaces. In *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*, pages 565–581, 2010.

39 Stanislaw J. Szarek. Convexity, complexity, and high dimensions. In *Proceedings of the International Congress of Mathematicians, Madrid, Spain*, pages 1599–1621. European Mathematical Society, 2006.

A Proof of Lemma 15

Fix a typical sequence $z = (z_1, \ldots, z_q)$. Our goal is to show that the total variation distance of $\mathcal{E}_{\mathsf{yes}}(z)$ and $\mathcal{E}^*_{\mathsf{no}}(z)$ is $o(1)$. For this purpose, we define a distribution $\mathcal{F}$ over pairs $(\mathbf{b}, \mathbf{d})$ of strings in $\{0,1\}^q$ (as a coupling of $\mathcal{E}_{\mathsf{yes}}(z)$ and $\mathcal{E}^*_{\mathsf{no}}(z)$), where the marginal distribution of $\mathbf{b}$ as $(\mathbf{b}, \mathbf{d}) \leftarrow \mathcal{F}$ is the same as $\mathcal{E}_{\mathsf{yes}}(z)$ and the marginal distribution of $\mathbf{d}$ is the same as $\mathcal{E}^*_{\mathsf{no}}(z)$. Our goal follows by establishing

$$\Pr_{(\mathbf{b},\mathbf{d})\leftarrow\mathcal{F}}\big[\mathbf{b} \neq \mathbf{d}\big] = o(1). \tag{17}$$

To define $\mathcal{F}$, we use $\mathbf{M}$ to denote the $q \times N$ $\{0,1\}$-valued random matrix derived from z and $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$ (recall that $\boldsymbol{S}$ is the intersection of N random halfspaces $\boldsymbol{h}_j$, $j \in [N]$): the (i,j)th entry $\mathbf{M}_{i,j}$ of $\mathbf{M}$ is 1 if $\boldsymbol{h}_j(z_i) = 1$ (i.e., $z_i \in \boldsymbol{h}_j$) and is 0 otherwise. We use $\mathbf{M}_{i,*}$ to denote the ith row of $\mathbf{M}$, $\mathbf{M}_{*,j}$ to denote the jth column of $\mathbf{M}$, and $\mathbf{M}^{(i)}$ to denote the $i \times N$ sub-matrix of $\mathbf{M}$ that consists of the first i rows of $\mathbf{M}$. (We note that $\mathbf{M}$ is derived from $\boldsymbol{S}$ and they are defined over the same probability space. So we may consider the (conditional) distribution of $\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}$ conditioning on an event involving $\mathbf{M}$, and we may consider the conditional distribution of $\mathbf{M}$ conditioning on an event involving $\boldsymbol{S}$.)

We now define $\mathcal{F}$. A pair $(\mathbf{b}, \mathbf{d}) \leftarrow \mathcal{F}$ is drawn using the following randomized procedure. The procedure has q rounds and generates the ith bits $\mathbf{b}_i$ and $\mathbf{d}_i$ in the ith round:

1. In the first round, we draw a random real number $\mathbf{r}_1$ from $[0,1]$ uniformly at random. We set $\mathbf{b}_1 = 1$ if $\mathbf{r}_1 \leq \mathbf{Pr}_{\boldsymbol{S}\leftarrow\mathcal{D}_{\mathsf{yes}}}[\boldsymbol{S}(z_1) = 1]$ and set $\mathbf{b}_1 = 0$ otherwise. We then set $\mathbf{d}_1 = 1$ if $\mathbf{r}_1 \leq \rho(\|z_1\|)$ and set $\mathbf{d}_1 = 0$ otherwise. (Note that for the first round, the two thresholds are indeed the same so we always have $\mathbf{b}_1 = \mathbf{d}_1$.) At the end of the first round, we also draw a vector $\mathbf{N}_{1,*}$ according to the distribution of $\mathbf{M}_{1,*}$ conditioning on $\boldsymbol{S}(z_1) = \mathbf{b}_1$.
2. In the ith round, for each i from 2 to q, we draw a random real number $\mathbf{r}_i$ from $[0,1]$ uniformly at random. We set $\mathbf{b}_i = 1$ if we have
$$\mathbf{r}_i \leq \mathop{\mathbf{Pr}}_{\boldsymbol{S}\leftarrow\mathcal{D}_{\mathsf{yes}}}\left[\boldsymbol{S}(z_i) = 1 \,\middle|\, \mathbf{M}^{(i-1)} = \mathbf{N}^{(i-1)}\right]$$
and set $\mathbf{b}_i = 0$ otherwise. We then set $\mathbf{d}_i = 1$ if $\mathbf{r}_i \leq \rho(\|z_i\|)$ and set $\mathbf{d}_i = 0$ otherwise. At the end of the ith round, we also draw a vector $\mathbf{N}_{i,*}$ according to the distribution of $\mathbf{M}_{i,*}$ conditioning on $\mathbf{M}^{(i-1)} = \mathbf{N}^{(i-1)}$ and $\boldsymbol{S}(z_i) = \mathbf{b}_i$.

It is clear that the marginal distributions of $\mathbf{b}$ and $\mathbf{d}$, as $(\mathbf{b}, \mathbf{d}) \leftarrow \mathcal{F}$, are indeed the same as $\mathcal{E}_{\mathsf{yes}}$ and $\mathcal{E}^*_{\mathsf{no}}$ respectively.

To prove (17), we introduce the following notion of *nice* and *bad* matrices.

▶ **Definition 18.** We say an $i \times N$ $\{0,1\}$-valued matrix M, for some $i \in [q]$, is *nice* if
1. M has at most $\sqrt{N}$ many 0-entries; and
2. Each column of M has at most one 0-entry.

We say M is *bad* otherwise.

We prove the following two lemmas and use them to prove (17).

▶ **Lemma 19.** $\mathbf{Pr}_{\boldsymbol{S}\leftarrow\mathcal{D}_{\mathsf{yes}}}\big[\mathbf{M}$ *is bad*$\big] = o(1/q)$.

Note that when $\mathbf{M}$ is nice, we have by definition that $\mathbf{M}^{(i)}$ is also nice for every $i \in [q]$.

▶ **Lemma 20.** *For any nice $(i-1) \times N$ $\{0,1\}$-valued matrix $M^{(i-1)}$, we have*
$$\mathop{\mathbf{Pr}}_{\boldsymbol{S}\leftarrow\mathcal{D}_{\mathsf{yes}}}\left[\boldsymbol{S}(z_i) = 1 \,\middle|\, \mathbf{M}^{(i-1)} = M^{(i-1)}\right] = \rho(\|z_i\|) \pm o(1/q). \tag{18}$$

Before proving Lemma 19 and 20, we first use them to prove (17). Let $\mathbf{I}_i$ denote the indicator random variable that is 1 if $(\mathbf{b}, \mathbf{d}) \leftarrow \mathcal{E}$ has $\mathbf{b}_i \neq \mathbf{d}_i$ and is 0 otherwise, for each $i \in [q]$. Then (17) is bounded from above by $\sum_{i\in[q]} \mathbf{Pr}[\mathbf{I}_i = 1]$. To bound each $\mathbf{Pr}[\mathbf{I}_i = 1]$ we split the event into
$$\sum_{M^{(i-1)}} \mathbf{Pr}\left[\mathbf{N}^{(i-1)} = M^{(i-1)}\right] \cdot \mathbf{Pr}\left[\mathbf{I}_i = 1 \,\middle|\, \mathbf{N}^{(i-1)} = M^{(i-1)}\right],$$
where the sum is over all $(i-1) \times N$ $\{0,1\}$-valued matrices $M^{(i-1)}$, and further split the sum into two sums over nice and bad matrices $M^{(i-1)}$. As $\mathbf{N}^{(i-1)}$ has the same distribution as $\mathbf{M}^{(i-1)}$, it follows from Lemma 19 (and the fact that $\mathbf{M}$ is bad when $\mathbf{M}^{(i-1)}$ is bad) that the sum over bad $M^{(i-1)}$ is at most $o(1/q)$. On the other hand, it follows from Lemma 20 that the sum over nice $M^{(i-1)}$ is $o(1/q)$. As a result, we have $\mathbf{Pr}[\mathbf{I}_i = 1] = o(1/q)$ and thus, $\sum_{i\in[q]} \mathbf{Pr}[\mathbf{I}_i = 1] = o(1)$.

We prove Lemmas 19 and 20 in the rest of the section.

Proof of Lemma 19. We show that the probability of $\mathbf{M}$ violating each of the two conditions in the definition of nice matrices is $o(1/q)$. The lemma then follows by a union bound.

For the first condition, since z is typical the probability of $\mathbf{M}_{i,j} = 0$ is
$$\mathrm{fsa}\big(\mathrm{cover}(z_i)\big) \leq e^{-0.49r^2}.$$

By linearity of expectation, the expected number of 0-entries in $\mathbf{M}$ is at most

$$qN \cdot e^{-0.49\,r^2} = o(\sqrt{N}/q),$$

using $e^{r^2/2} \ge N/n$, $N = 2^{\sqrt{n}}$ and $q = 2^{0.01\sqrt{n}}$. It follows directly from Markov's inequality that the probability of $\mathbf{M}$ having more than $\sqrt{N}$ many 0-entries is $o(1/q)$.

For the second condition, again since z is typical, the probability of $\mathbf{M}_{i,j} = \mathbf{M}_{i',j} = 1$ is

$$\mathrm{fsa}\big(\mathrm{cover}(z_i) \cap \mathrm{cover}(z_i')\big) \le e^{-0.96\,r^2}.$$

By a union bound, the probability of $\mathbf{M}_{i,j} = \mathbf{M}_{i',j} = 1$ for some i, i', j is at most

$$q^2 N \cdot e^{-0.96r^2} = o(1/q).$$

This finishes the proof of the lemma. ◀

Finally we prove Lemma 20. Fix a nice $(i-1) \times N$ matrix M (we henceforth omit the superscript $(i-1)$ since the number of rows of M is fixed to be $i-1$). Recall that $\boldsymbol{S}(z_i) = 1$ if and only if $\boldsymbol{h}_j(z_i) = 1$ for all $j \in [N]$. As a result, we have

$$\Pr_{\boldsymbol{S} \leftarrow \mathcal{D}_{\mathsf{yes}}}\Big[\boldsymbol{S}(z_i) = 1 \,\big|\, \mathbf{M}^{(i-1)} = M\Big] = \prod_{j \in [N]} \Pr_{\boldsymbol{h}_j}\Big[\boldsymbol{h}_j(z_i) = 1 \,\big|\, \mathbf{M}^{(i-1)}_{*,j} = M_{*,j}\Big].$$

On the other hand, letting $\tau = \mathrm{fsa}(\mathrm{cover}(z_i)) = \mathrm{cap}(r/\|z_i\|)$, we have $\rho(\|z_i\|) = (1-\tau)^N$.

In the next two claims we compare

$$\Pr_{\boldsymbol{h}_j}\Big[\boldsymbol{h}_j(z_i) = 1 \,\big|\, \mathbf{M}^{(i-1)}_{*,j} = M_{*,j}\Big]$$

with $1-\tau$ for each $j \in [N]$ and show that they are very close. The first claim works on $j \in [N]$ with no 0-entry in $M_{*,j}$ and the second claim works on $j \in [N]$ with one 0-entry in $M_{*,j}$. (These two possibilities cover all $j \in [N]$ since the matrix M is nice.) Below we omit $\mathbf{M}^{(i-1)}_{*,j}$ in writing the conditional probabilities.

▶ **Claim 21.** *For each $j \in [N]$ with no 0-entry in the jth column $M_{*,j}$, we have*

$$\Pr_{\boldsymbol{h}_j}\Big[\boldsymbol{h}_j(z_i) = 1 \,\big|\, M_{*,j}\Big] = (1-\tau)\left(1 \pm \frac{o(1)}{qN}\right).$$

Proof. Let δ be the probability of $\boldsymbol{h}_j(z_i) = 0$ conditioning on $M_{*,j}$ (which is all-1). Then

$$\delta = \frac{\mathrm{fsa}\left(\mathrm{cover}(z_i) - \bigcup_{j<i} \mathrm{cover}(z_j)\right)}{1 - \mathrm{fsa}\left(\bigcup_{j<i} \mathrm{cover}(z_j)\right)}.$$

Using $e^{-0.51\,r^2} \le \mathrm{fsa}(\mathrm{cover}(z_j)) \le e^{-0.49\,r^2}$ and $\mathrm{fsa}(\mathrm{cover}(z_i) \cap \mathrm{cover}(z_j)) \le e^{-0.96\,r^2}$, we have

$$\delta \le \frac{\tau}{1 - q \cdot e^{-0.49\,r^2}} < \tau(1 + 2q \cdot e^{-0.49\,r^2}) = \tau + 2\tau q \cdot e^{-0.49\,r^2}.$$

Using $\tau \le e^{-0.49\,r^2}$ and $e^{r^2/2} \ge N/n$, we have

$$1 - \delta \ge 1 - \tau - 2\tau q \cdot e^{-0.49\,r^2} \ge 1 - \tau - o\big(1/(qN)\big) \ge (1-\tau)\big(1 - o(1/(qN))\big).$$

On the other hand, we have $\delta \ge \tau - q \cdot e^{-0.96\,r^2}$ and thus,

$$1 - \delta \le 1 - \tau + q \cdot e^{-0.96\,r^2} \le 1 - \tau + o\big(1/(qN)\big) = (1-\tau)\big(1 + o(1/(qN))\big).$$

This finishes the proof of the claim. ◀

▶ **Claim 22.** *For each $j \in [N]$ with one 0-entry in the jth column $M_{*,j}$, we have*

$$\Pr_{\boldsymbol{h}_j}\Big[\boldsymbol{h}_j(z_i) = 1 \,\Big|\, M_{*,j}\Big] \ge 1 - O\big(e^{-0.45\,r^2}\big).$$

Proof. Let i' be the point with $M_{i',j} = 1$ and δ be the conditional probability of $\boldsymbol{h}_j(z_i) = 0$. Then we have

$$\delta \le \frac{\mathrm{fsa}\big(\mathrm{cover}(z_i) \cap \mathrm{cover}(z_{i'})\big)}{\mathrm{fsa}\left(\mathrm{cover}(z_i') - \bigcup_{j<i:\, j\neq i'} \mathrm{cover}(z_j)\right)} \le \frac{e^{-0.96\,r^2}}{e^{-0.51\,r^2} - q \cdot e^{-0.96\,r^2}} = O\big(e^{-0.45\,r^2}\big),$$

by our choice of q. This finishes the proof of the claim. ◀

We combine the two claims to prove Lemma 20.

Proof of Lemma 20. Let h be the number of 0-entries in M. We have $h \le \sqrt{N}$ since M is nice. By Claims 21, the conditional probability of $\boldsymbol{S}(z_i) = 1$ is at most

$$\begin{aligned}\left((1-\tau)\left(1 + o\left(\frac{1}{qN}\right)\right)\right)^{N-h} &= \rho(\|z_i\|) \cdot \frac{1}{(1-\tau)^h} \cdot \left(1 + o\left(\frac{1}{qN}\right)\right)^{N-h} \\ &\le \rho(\|z_i\|) \cdot (1+2\tau)^h \cdot \left(1 + o\left(\frac{1}{qN}\right)\right)^{N} \\ &\le \rho(\|z_i\|) \cdot \exp\big(2\tau h + o(1/q)\big) \\ &= \rho(\|z_i\|) \cdot \exp\big(o(1/q)\big) = \rho(\|z_i\|) + o(1/q).\end{aligned}$$

Similarly, the conditional probability of $\boldsymbol{S}(z_i) = 1$ is at least

$$\begin{aligned}&\left((1-\tau)\left(1 - o\left(\frac{1}{qN}\right)\right)\right)^{N-h} \left(1 - O\left(e^{-0.45r^2}\right)\right)^h \\ &\qquad \ge \rho(\|z_i\|) \cdot \left(1 - o\left(\frac{1}{qN}\right)\right)^{N-h} \left(1 - O\left(e^{-0.45r^2}\right)\right)^h \\ &\qquad \ge \rho(\|z_i\|) \cdot \big(1 - o(1/q)\big) \ge \rho(\|z_i\|) - o(1/q).\end{aligned}$$

This finishes the proof of the lemma. ◀

Adaptivity Is Exponentially Powerful for Testing Monotonicity of Halfspaces[*][†]

Xi Chen[1], Rocco A. Servedio[2], Li-Yang Tan[3], and Erik Waingarten[4]

1 **Columbia University, New York, NY, USA**
`xichen@cs.columbia.edu`

2 **Columbia University, New York, NY, USA**
`rocco@cs.columbia.edu`

3 **Toyota Technological Institute, Chicago, IL, USA**
`liyang@cs.columbia.edu`

4 **Columbia University, New York, NY, USA**
`eaw@cs.columbia.edu`

Abstract

We give a $\text{poly}(\log n, 1/\epsilon)$-query adaptive algorithm for testing whether an unknown Boolean function $f\colon \{-1,1\}^n \to \{-1,1\}$, which is promised to be a halfspace, is monotone versus ϵ-far from monotone. Since non-adaptive algorithms are known to require almost $\Omega(n^{1/2})$ queries to test whether an unknown halfspace is monotone versus far from monotone, this shows that adaptivity enables an exponential improvement in the query complexity of monotonicity testing for halfspaces.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases property testing, linear threshold functions, monotonicity, adaptivity

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.38

1 Introduction

Monotonicity testing has been a touchstone problem in property testing for more than fifteen years [17, 23, 19, 22, 21, 3, 1, 24, 29, 6, 8, 27, 10, 11, 12, 7, 14, 25, 13, 4, 16], with many exciting recent developments leading to a greatly improved understanding of the problem in just the past few years. The seminal work of [23] introduced the problem and gave an $O(n/\epsilon)$-query algorithm that tests whether an unknown and arbitrary function $f\colon \{-1,1\}^n \to \{-1,1\}$ is monotone versus ϵ-far from every monotone function. While steady progress followed for non-Boolean functions and for functions over other domains, the first improved algorithm for Boolean-valued functions over $\{-1,1\}^n$ was only achieved in [10], who gave a $\tilde{O}(n^{7/8}) \cdot \text{poly}(1/\epsilon)$-query non-adaptive testing algorithm. A slightly improved $\tilde{O}(n^{5/6}) \cdot \text{poly}(1/\epsilon)$-query non-adaptive algorithm was given by [14], and subsequently [25] gave a $\tilde{O}(n^{1/2}) \cdot \text{poly}(1/\epsilon)$-query non-adaptive algorithm.

On the lower bounds side, the fundamental class of *halfspaces* has played a major role in non-adaptive lower bounds for monotonicity testing to date. We discuss lower bounds for two-sided error monotonicity testing of Boolean-valued functions over $\{-1,1\}^n$, and refer the

[*] A full version of the paper is available at `https://arxiv.org/abs/1706.05556`.

[†] X. C. was supported by NSF grants CCF-1149257 and CCF-1423100. R. A. S. was supported by NSF grants CCF-1420349 and CCF-1563155. L.-Y. T. was supported by NSF grant CCF-1563122. E. W. was supported by the NSF Graduate Research Fellowship under Grant No. DGE-16-44869.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 38; pp. 38:1–38:21
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

reader to the above references for lower bounds on other variants of the monotonicity testing problem. The first (two-sided) lower bound was established by Fischer et al [22], who used a slight variant of the majority function to give an $\Omega(\log n)$ lower bound for non-adaptive monotonicity testing. More recently, the lower bound of [13], strengthening [14], shows that for any constant $\delta > 0$, there is a constant $\epsilon = \epsilon(\delta) > 0$ such that $\Omega(n^{1/2-\delta})$ non-adaptive queries are required to distinguish whether a Boolean function f – which is promised to be a halfspace – is monotone or ϵ-far from every monotone function. Together with the $\tilde{O}(n^{1/2}) \cdot \mathrm{poly}(1/\epsilon)$-query non-adaptive monotonicity testing algorithm of [25], this shows that halfspaces are "as hard as the hardest functions" to non-adaptively test for monotonicity. Halfspaces are also commonly referred to as "linear threshold functions" or LTFs; for brevity we shall subsequently refer to them as LTFs.

The role of adaptivity

While the above results largely settle the query complexity of non-adaptive monotonicity testing, the situation is less clear when adaptive algorithms are allowed. More generally, the power of adaptivity in property testing is not yet well understood, despite being a natural and important question.[1] A recent breakthrough result of Belovs and Blais [4] gives a $\tilde{\Omega}(n^{1/4})$ lower bound on the query complexity of adaptive algorithms that test whether $f\colon \{-1,1\}^n \to \{-1,1\}$ is monotone versus ϵ-far from monotone, for some absolute constant $\epsilon > 0$. This result was then improved by [16] to $\tilde{\Omega}(n^{1/3})$. [4] also shows that when f is promised to be an "extremely regular" LTF, with regularity parameter at most $O(1)/\sqrt{n}$, then $\log n + O_\epsilon(1)$ adaptive queries suffice. (We define the "regularity" of an LTF in part (a) of Definition 2 below. Here we note only that every n-variable LTF has regularity between $1/\sqrt{n}$ and 1, so $O(1)/\sqrt{n}$-regular LTFs are "extremely regular" LTFs.)

A very compelling question is whether adaptivity helps for monotonicity testing of Boolean functions: can adaptive algorithms go below the [13] $\Omega(n^{1/2-\delta})$-query lower bound for non-adaptive algorithms? While we do not know the answer to this question for general Boolean functions[2], in this work we give a strong positive answer in the case of LTFs, generalizing the upper bound of [4] from "extremely regular" LTFs to arbitrary unrestricted LTFs. The main result of this work is an adaptive algorithm with one-sided error that can test any LTF for monotonicity using $\mathrm{poly}(\log n, 1/\epsilon)$ queries:

▶ **Theorem 1** (Main). *There is a* $\mathrm{poly}(\log n, 1/\epsilon)$*-query*[3] *adaptive algorithm with the following property: given* $\epsilon > 0$ *and black-box access to an unknown LTF* $f\colon \{-1,1\}^n \to \{-1,1\}$*,*

- *If f is monotone then the algorithm outputs "monotone" with probability* 1*;*
- *If f is ϵ-far from every monotone function then the algorithm outputs "non-monotone" with probability at least* 2/3*.*

[1] For monotonicity testing of functions $f\colon [n]^2 \to \{0,1\}$, Berman et al. [5] showed that adaptive algorithms are strictly more powerful than non-adaptive ones (by a factor of $\log 1/\epsilon$). For unateness testing of real-valued functions $f\colon \{0,1\}^n \to \mathbb{R}$, a natural generalization of monotonicity, [2] showed that adaptivity helps by a logarithmic factor. We remark that for another touchstone class in property testing, the class of Boolean juntas, it was only very recently shown [30, 15] that adaptive algorithms are strictly more powerful than non-adaptive algorithms.

[2] For very special functions such as truncated anti-dictators, it is known [22] that adaptive algorithms are known to be much more efficient than nonadaptive algorithms ($O(\log n)$ versus $\Omega(\sqrt{n})$ queries) in finding a violation to monotonicity.

[3] See Theorem 26 of Section 5 for a detailed description of the algorithm's query complexity; we have made no effort to optimize the particular polynomial dependence on $\log n$ and $1/\epsilon$ that the algorithm achieves.

Recalling that the $\Omega(n^{1/2-\delta})$ non-adaptive lower bound from [13] is proved using LTFs as both the yes- and no- functions, Theorem 1 shows that adaptive algorithms are exponentially more powerful than non-adaptive algorithms for testing monotonicity of LTFs. Together with the $\tilde{\Omega}(n^{1/3})$ adaptive lower bound from [16], it also shows that LTFs are exponentially easier to test for monotonicity than general Boolean functions using adaptive algorithms.

1.1 A very high-level overview of the algorithm

The adaptive algorithm of [4] for testing monotonicity of "extremely regular" LTFs is essentially based on a simple binary search over the hypercube $\{-1,1\}^n$ to find an anti-monotone edge[4]. [4] succeeds in analyzing such an algorithm, taking advantage of some of the nice structural properties of regular LTFs, but it is not clear how to carry out such an analysis for general LTFs.

To deal with general LTFs, our algorithm is more involved and employs an iterative stage-wise approach, running for up to $O(\log n)$ stages. Entering the $(t+1)$-th stage, the algorithm maintains a restriction $\rho^{(t)}$ that fixes some of the input variables to f, and in the $(t+1)$-th stage the algorithm queries $f_{\rho^{(t)}}$, where we write $f_{\rho^{(t)}}$ to denote the function f after the restriction $\rho^{(t)}$. At a very high level, in the $(t+1)$-th stage the algorithm either

(i) Obtains definitive evidence (in the form of an anti-monotone edge) that $f_{\rho^{(t)}}$, and hence f, is not monotone. In this case the algorithm halts and outputs "non-monotone." Or, it

(ii) Extends the restriction $\rho^{(t)}$ to obtain $\rho^{(t+1)}$. This is done by fixing a random subset of the variables of expected density 1/2 that are not fixed under $\rho^{(t)}$, and possibly some additional variables, in such a way as to maintain an invariant described later. Or, it

(iii) Fails to achieve (i) or (ii), which we show is very unlikely to happen. In this case the algorithm simply halts and outputs "monotone."

We describe the invariant of $\rho^{(t)}$ maintained in Case (ii) in Section 1.2. One of its implications in particular is that $f_{\rho^{(t)}}$ is ϵ'-far from monotone, where ϵ' has a polynomial dependence on ϵ. As a result, when the number of surviving variables under $\rho^{(t^*)}$ at the beginning of a stage t^* is at most $\mathrm{poly}(\log n)$, the algorithm can run the simple "edge tester" of [23] on $f_{\rho^{(t^*)}}$ to find an anti-monotone edge with high probability. Although the "edge tester" has query complexity linear in the number of variables, this is affordable since $f_{\rho^{(t^*)}}$ only has $\mathrm{poly}(\log n)$ many variables left. Case (ii) ensures that there are at most $O(\log n)$ stages overall. We will also see that each stage makes at most $\mathrm{poly}(\log n, 1/\epsilon)$ queries; hence the overall query complexity is $\mathrm{poly}(\log n, 1/\epsilon)$.

1.2 A more detailed overview of the algorithm and why it works

In this section we give a more detailed overview of the algorithm and a high-level sketch of its analysis. The algorithm only outputs "non-monotone" if it identifies an anti-monotone edge, so it will correctly output "monotone" on every monotone f with probability 1. Hence, establishing correctness of the algorithm amounts to showing that if f is an LTF that is ϵ-far from monotone, then with high probability the algorithm will output "non-monotone" when it runs on f. Thus, for the remainder of this section, $f(x) = \mathrm{sign}(w_1x_1 + \cdots + w_nx_n - \theta)$ should be viewed as being an LTF that is ϵ-far from monotone.

A crucial notion for understanding the algorithm is that of a (τ,γ,λ)-*non-monotone LTF.*

[4] A *bi-chromatic* edge of $f\colon \{-1,1\}^n \to \{-1,1\}$ is a pair (x,y) of points such that $x,y \in \{-1,1\}^n$ differ at exactly one coordinate and satisfy $f(x) \neq f(y)$. An *anti-monotone* edge of f is a bi-chromatic edge (x,y) that also satisfies $x_i = -1, y_i = 1$ for some $i \in [n]$ and $f(x) = 1, f(y) = -1$.

▶ **Definition 2.** Given an LTF $f\colon \{-1,1\}^S \to \{-1,1\}$ of the form $f(x) = \mathrm{sign}(w \cdot x - \theta)$ over a set of variables S, we say it is a (τ, γ, λ)-*non-monotone LTF with respect to the weights* w if it satisfies the following three properties:

(a) f is τ-*weight-regular*[5] with respect to w, i.e.,

$$\max_{i \in S} |w_i| \le \tau \cdot \sqrt{\sum_{j \in S} w_j^2};$$

(b) f is γ-*balanced*, i.e., $\left|\mathbf{E}_{\mathbf{x} \in \{-1,1\}^n}[f(\mathbf{x})]\right| \le 1 - \gamma$; and

(c) f has λ-*significant squared negative weights in* w, i.e.,

$$\frac{\sum_{i \in S: w_i < 0} (w_i)^2}{\sum_{i \in S} (w_i)^2} \ge \lambda.$$

Looking ahead, an insight that underlies this definition (as well as our algorithm) is that, when $f = \mathrm{sign}(w \cdot x - \theta)$ is a weight-regular LTF that is far from monotone, f must satisfy (c) above for some large value of λ (see Lemma 12 for a precise formulation). The converse also holds, i.e., an LTF that satisfies all three conditions above must be ϵ-far from monotone for some large value of ϵ (see Lemma 13). This is indeed the reason why we call such functions (τ, γ, λ)-*non-monotone* LTFs. An additional motivation for the regularity condition (a) is that, when f satisfies (c) for some value $\lambda \gg \tau$ (the parameter in (a)), a random restriction ρ (that randomly fixes half of the variables to uniform values from $\{-1,1\}$) would have f_ρ still satisfy (c) with essentially the same λ. The balance condition (b), on the other hand, may be viewed as a technical condition that makes it possible for our various subroutines to work efficiently and correctly; we note that if f is not γ-balanced, then f is trivially $(\gamma/2)$-close to either the monotone function 1 or the monotone function -1.

With Definition 2 in hand, we proceed to a more detailed overview of the algorithm (still at a rather conceptual level). The algorithm takes as input black-box access to $f\colon \{-1,1\}^n \to \{-1,1\}$ and a parameter $\epsilon > 0$. We remind the reader that in the subsequent discussion f should be viewed as an ϵ-far-from-monotone LTF. For the analysis of the algorithm, we also assume that f takes the form of $f(x) = \mathrm{sign}(w_1 x_1 + \cdots + w_n x_n - \theta)$, for some unknown (but fixed[6]) weight vector w and threshold θ. They are unknown to the algorithm and will be used in the analysis only.

Our algorithm has two main phases: first an *initialization* phase, and then the phase consisting of the *main procedure.*

Initialization. The algorithm runs an initialization procedure `Regularize-and-Balance`. Roughly speaking, it with high probability either identifies f as a non-monotone LTF by finding an anti-monotone edge and halts, or constructs a restriction $\rho^{(0)}$ such that $f_{\rho^{(0)}}$ becomes a $(\tau, \gamma, \lambda_0)$-non-monotone LTF for suitable parameters τ, γ, λ_0, with $\tau = \mathrm{poly}(1/\log n, \epsilon)$, $\gamma = \epsilon$, $\lambda_0 = \mathrm{poly}(\epsilon)$ and $\tau \ll \lambda_0$. In the latter case the algorithm continues with $f_{\rho^{(0)}}$.

Main Procedure. As sketched earlier in Section 1.1 the main procedure operates in a sequence of $O(\log n)$ stages. In its $(t+1)$th stage, it operates on the restricted function

[5] Our terminology "weight-regular" means the same thing as [4]'s "regular." We use the terminology "weight-regular" to distinguish it from the different notion of "Fourier-regularity" which we also require, see Section 2.2.

[6] Note that (w, θ) is not unique for a given f. We pick such a pair and stick to it throughout the analysis.

$f_{\rho^{(t)}}$ which is assumed to be a $(\tau, \gamma, \lambda_t)$-non-monotone LTF, and with high probability either identifies f as non-monotone and halts, or constructs an extension $\rho^{(t+1)}$ of the restriction $\rho^{(t)}$ such that $f_{\rho^{(t+1)}}$ remains $(\tau, \gamma, \lambda_{t+1})$-non-monotone (for some parameter λ_{t+1} that is only slightly smaller than λ_t) while the number of free variables in $\rho^{(t+1)}$ drops by a constant factor.

To describe each stage in more detail, we need the following notation for restrictions. Given a restriction $\rho \in \{-1, 1, *\}^{[n]}$, we use $\textsc{stars}(\rho)$ to denote the set of indices that are not fixed in ρ, i.e., the set of i such that $\rho(i) = *$. Given $f \colon \{-1, 1\}^n \to \{-1, 1\}$ of the form $f(x) = \text{sign}(\sum w_i x_i - \theta)$, we let $f_\rho \colon \{-1, 1\}^{\textsc{stars}(\rho)} \to \{-1, 1\}$ denote the function f after the restriction ρ:

$$f_\rho(x) = \text{sign}\left(\textstyle\sum_{i \in \textsc{stars}(\rho)} w_i \cdot x_i + \sum_{j \notin \textsc{stars}(\rho)} w_j \cdot \rho(j) - \theta\right).$$

We stress than the weights of f_ρ remain w_i while the threshold is $\theta - \sum_{j \notin \textsc{stars}(\rho)} w_j \cdot \rho(j)$.

Now for the $(t + 1)$th stage, where $t = 0, 1, 2, \ldots$, the main procedure carries out the following sequence of steps (we defer discussion of how these steps are implemented to Section 5). Below for convenience we let g denote $f_{\rho^{(t)}}$, the function that the algorithm operates on in the $(t + 1)$th stage.

1. Draw a random subset $A_t \subset \textsc{stars}(\rho^{(t)})$, which consists of roughly half of its variables. Assuming that $\tau \ll \lambda_t$, we have that, with high probability, A_t partitions the positive and negative weights roughly evenly and the collection of weights of variables in $\textsc{stars}(\rho^{(t)}) \setminus A_t$ has λ_{t+1}-significant squared negative weights for some λ_{t+1} that is only slightly smaller than λ_t. (This also justifies the assumption of $\tau \ll \lambda_t$ at the beginning.)
2. Find a restriction $\rho' \in \{-1, 1, *\}^{\textsc{stars}(\rho^{(t)})}$ that fixes the variables in A_t in such a way that $g_{\rho'}$ is 0.96-balanced. The exact constant 0.96 here is not important as long as it is close enough to 1. Note that $g_{\rho'}$ is more balanced than g is promised to be (i.e., $(\gamma = \epsilon)$-balanced and we may assume that $\epsilon \le 0.5$). This helps in the last step of the stage. Our analysis shows that if g is $(\tau, \gamma, \lambda_t)$-non-monotone, then this step succeeds with high probability.
3. Find a set $H_t \subset \textsc{stars}(\rho^{(t)}) \setminus A_t$ that contains those variables x_i that have "high influence" in $g_{\rho'}$. Intuitively, H_t contains variables of $g_{\rho'}$ that violate the τ-weight-regularity condition; after its removal, the collection of weights of variables in $\textsc{stars}(\rho^{(t)}) \setminus (A_t \cup H_t)$ becomes τ-weight-regular again.
4. For each $i \in H_t$, find a bi-chromatic edge of $g_{\rho'}$ on the ith coordinate (this can be done efficiently because the variables in H_t all have high influence in $g_{\rho'}$), which reveals the sign of w_i. If an anti-monotone edge is found, halt and output "non-monotone;" otherwise, we know that the weight of every variable in H_t is positive.
5. Finally, find a restriction $\rho'' \in \{-1, 1, *\}^{\textsc{stars}(\rho^{(t)})}$, which extends ρ' and fixes the variables in $A_t \cup H_t$, such that $g_{\rho''}$ is γ-balanced. Our analysis shows that if g is $(\tau, \gamma, \lambda_t)$-non-monotone and $g_{\rho'}$ is 0.96-balanced, then this step succeeds with high probability. By Step 3, $g_{\rho''}$ is τ-weight-regular. In addition, $g_{\rho''}$ has λ_{t+1}-significant squared negative weights because of Step 1 and Step 4 (which makes sure that all variables in H_t have positive weights). At the end, we set $\rho^{(t+1)}$ to be the composition of $\rho^{(t)}$ and ρ'' and move on to the next stage.

To summarize, our analysis shows that if $f_{\rho^{(t)}}$ is $(\tau, \gamma, \lambda_t)$-non-monotone (entering the $(t + 1)$th stage) then with high probability the algorithm in the $(t + 1)$th stage either finds an anti-monotone edge and halts, or finds an extension $\rho^{(t+1)}$ of $\rho^{(t)}$ such that:

(i) The new function $f_{\rho^{(t+1)}}$ is $(\tau, \gamma, \lambda_{t+1})$-non-monotone (entering the $(t+2)$th stage), where the parameter λ_{t+1} is only slightly smaller than λ_t (more on this below); and

(ii) The number of surviving variables in $\rho^{(t+1)}$ is only about half of that of $\rho^{(t)}$.

This implies that, with high probability, the main procedure within $O(\log n)$ stages either finds an anti-monotone edge and returns the correct answer "non-monotone" or constructs a restriction $\rho^{(t)}$ such that $f_{\rho^{(t)}}$ is $(\tau, \gamma, \lambda_t)$-non-monotone and the number of surviving variables under $\rho^{(t)}$ is at most $m = \text{poly}(\log n, 1/\epsilon)$. For the latter case, our analysis (Lemma 13) together with the fact that λ_t drops only slightly in each stage show that $f_{\rho^{(t)}}$ remains $\epsilon' = \text{poly}(\epsilon)$-far from monotone. Thus, the algorithm concludes by running the "edge tester" from [23] to ϵ'-test the m-variable function $f_{\rho^{(t)}}$, which uses $O(m/\epsilon') = \text{poly}(\log n, 1/\epsilon)$ queries to $f_{\rho^{(t)}}$ and finds an anti-monotone edge with high probability. To summarize, when f is an LTF that is ϵ-far from monotone, our algorithm finds an anti-monotone edge and outputs "non-monotone" with high probability. As discussed earlier at the beginning of Section 1.2 about its one-sideness, the correctness of the algorithm follows.

1.3 Relation to previous work

We have already discussed how our main result, Theorem 1, relates to the recent upper and lower bounds of [25, 13, 4] for monotonicity testing. At the level of techniques, several aspects of our algorithm are reminiscent of some earlier work in property testing of Boolean functions and probability distributions as we describe below.

At a high level, the $\text{poly}(1/\epsilon)$-query algorithm of [26] for testing whether a function is an LTF identifies high-influence variables and "deals with them separately" from other variables, as does our algorithm. The more recent algorithm of [28], for testing whether a function is a signed majority function, like our algorithm proceeds in a series of stages which successively builds up a restriction by fixing more and more variables. Like our algorithm the [28] algorithm makes only $\text{poly}(\log n, 1/\epsilon)$ adaptive queries, but there are many differences both between the two algorithms and between their analyses. To briefly note a few of these differences, the [28] algorithm has two-sided error while our algorithm has one-sided error; the former also heavily leverages both the very "rigid" structure of the degree-1 Fourier coefficients of any signed majority function and the near-perfect balancedness of any signed majority function between the two outputs 1 and -1, neither of which hold in our setting. Finally, we note that the general approach of iteratively selecting and retaining a random subset of the remaining "live" elements, then doing some additional pruning to identify, check, and discard a small number of "heavy" elements, then proceeding to the next stage is reminiscent of the Approx-Eval-Simulator procedure of [9], which deals with testing probability distributions in the "conditional sampling" model.

1.4 Organization

In Section 2 we recall the necessary background concerning monotonicity, LTFs, and restrictions, and state a few useful algorithmic and structural results from prior work. In Section 3 we establish several new structural results about "regular" LTFs: we first show that its distance to monotonicity corresponds (approximately) to its total amount of squared negative coefficient weights; we also prove that its distance to monotonicity is preserved under a random restriction to a set of its non-decreasing variables. In Section 4 we present and analyze some simple algorithmic subroutines that will be used to identify high influence variables and check that they are non-decreasing. Finally in Section 5, we give a detailed description of our overall algorithm for testing monotonicity of LTFs, and prove its correctness, establishing our main result (Theorem 1).

2 Background

We write $[n]$ for $\{1,\dots,n\}$, and use boldface letters (e.g. $\mathbf{x}$ and $\mathbf{X}$) to denote random variables. We briefly recall some basic notions. A function $f\colon \{-1,1\}^n \to \{-1,1\}$ is *monotone* (short for "monotone non-decreasing") if $x \preceq y$ implies $f(x) \le f(y)$, where "$x \preceq y$" means that $x_i \le y_i$ for all $i \in [n]$. A function f is *unate* if there is a bit vector $a \in \{-1,1\}^n$ such that $f(a_1x_1,\dots,a_nx_n)$ is monotone. It is well known that every LTF (defined below) is unate.

We measure distance between functions $f, g\colon \{-1,1\}^n \to \{-1,1\}$ with respect to the uniform distribution, so we say that f and g are *ϵ-close* if

$$\mathrm{dist}(f,g)\colon = \Pr_{\mathbf{x}\in\{-1,1\}^n}\big[f(\mathbf{x}) \neq g(\mathbf{x})\big] \le \epsilon,$$

and that f and g are *ϵ-far* otherwise. A function f is *ϵ-far from monotone* if it is ϵ-far from every monotone function g. We write $\mathrm{dist}(f, \textsc{Mono})$ to denote the minimum value of $\mathrm{dist}(f,g)$ over all monotone functions g. Throughout the paper all probabilities and expectations are with respect to the uniform distribution over $\{-1,1\}^n$ unless otherwise indicated. As indicated in Definition 2, we say that a $\{-1,1\}$-valued function f is *γ-balanced* if

$$\left| \mathop{\mathbf{E}}_{\mathbf{x}\in\{-1,1\}^n}[f(\mathbf{x})] \right| \le 1-\gamma.$$

A function $g\colon \{-1,1\}^n \to \{-1,1\}$ is a *junta* over $S \subseteq [n]$ if g depends only on the coordinates in S. We say f is *ϵ-close* to a *junta* over S if f is ϵ-close to g for some g that is a junta over S.

2.1 LTFs and weight-regularity

A function $f\colon \{-1,1\}^n \to \{-1,1\}$ is an *LTF* (also commonly referred to as a *halfspace*) if there exist real weights $w_1,\dots,w_n \in \mathbb{R}$ and a real threshold $\theta \in \mathbb{R}$ such that

$$f(x) = \begin{cases} 1 & \text{if } w_1x_1 + \cdots + w_nx_n \ge \theta, \\ -1 & \text{if } w_1x_1 + \cdots + w_nx_n < \theta. \end{cases}$$

We say that $w = (w_1,\dots,w_n)$ are the *weights* and θ the *threshold* of the LTF, and we say that (w,θ) *represents* the LTF f, or simply that $f(x)$ is the LTF given by $\mathrm{sign}(w\cdot x - \theta)$. Note that for any LTF f there are in fact infinitely many pairs (w,θ) that represent f; we fix a particular pair (w,θ) for each n-variable LTF f and work with it in what follows.

An important notion in our arguments is that of *weight-regularity.* As indicated in Definition 2, given a weight vector $w \in \mathbb{R}^n$, we say that w is *τ-weight-regular* if no more than a τ-fraction of the 2-norm of $w = (w_1,\dots,w_n)$ comes from any single coefficient w_i, i.e.,

$$\max_{i\in[n]} |w_i| \le \tau \cdot \sqrt{w_1^2 + \cdots + w_n^2}. \tag{1}$$

If we have fixed a representation (w,θ) for f such that w is τ-weight-regular, we frequently abuse the terminology and say that f is τ-weight-regular.

2.2 Fourier analysis of Boolean functions and Fourier-regularity

Given a function $f\colon \{-1,1\}^n \to \mathbb{R}$, we define its *Fourier coefficients* by $\hat{f}(S) = \mathbf{E}[f \cdot x_S]$ for each $S \subseteq [n]$, where x_S denotes $\prod_{i \in S} x_i$, and we have that $f(x) = \sum_S \hat{f}(S) \cdot x_S$. We will be particularly interested in f's *degree*-1 coefficients, i.e., $\hat{f}(S)$ for $|S| = 1$; we will write these as $\hat{f}(i)$ rather than $\hat{f}(\{i\})$. We recall *Plancherel's identity* $\langle f, g\rangle = \sum_S \hat{f}(S)\hat{g}(S)$, which has as a special case *Parseval's identity*, $\mathbf{E}_{\mathbf{x}}[f(\mathbf{x})^2] = \sum_S \hat{f}(S)^2$. It follows that every $f\colon \{-1,1\}^n \to \{-1,1\}$ has $\sum_S \hat{f}(S)^2 = 1$.

We further recall that, for any unate function $f\colon \{-1,1\}^n \to \{-1,1\}$ (and hence any LTF), we have $|\hat{f}(i)| = \mathbf{Inf}_i(f)$, where the *influence* of variable i on f is

$$\mathbf{Inf}_i(f) = \Pr_{\mathbf{x} \in \{-1,1\}^n} \big[f(\mathbf{x}) \neq f(\mathbf{x}^{\oplus i})\big],$$

where $x^{\oplus i}$ is the vector obtained from x by flipping coordinate i.

We say that $f\colon \{-1,1\}^n \to \{-1,1\}$ is *τ-Fourier-regular* if $\max_{i \in [n]} |\hat{f}(i)| \leq \tau$. Section 2.5 summarizes some relationships between weight-regularity and Fourier-regularity of LTFs.

2.3 Restrictions

A *restriction* ρ is an element of $\{-1,1,*\}^{[n]}$; we view ρ as a partial assignment to the n variables $x_1, \ldots, x_n$, where $\rho(i) = *$ indicates that variable x_i is unassigned. We write $\mathrm{supp}(\rho)$ to denote the set of indices i such that $\rho(i) \in \{-1,1\}$ and $\textsc{stars}(\rho)$ to denote the set of i such that $\rho(i) = *$ (and thus, $\textsc{stars}(\rho)$ is the complement of $\mathrm{supp}(\rho)$).

Given restrictions $\rho, \rho' \in \{-1,1,*\}^{[n]}$ we say that ρ' is an *extension* of ρ if $\mathrm{supp}(\rho) \subseteq \mathrm{supp}(\rho')$ and $\rho'(i) = \rho(i)$ for all $i \in \mathrm{supp}(\rho)$. If ρ and ρ' are restrictions with disjoint support we write $\rho\rho'$ to denote the *composition* of these two restrictions (that has support $\mathrm{supp}(\rho) \cup \mathrm{supp}(\rho')$).

2.4 Useful algorithmic tools from prior work

We recall some algorithmic tools for working with black-box functions $f\colon \{-1,1\}^n \to \{-1,1\}$.

Estimating sums of squares of degree-1 Fourier coefficients. We first recall Corollary 16 of [26] (slightly specialized to our context):

▶ **Lemma 3** (Corollary 16 [26]). *There is a procedure* `Estimate-Sum-of-Squares`(f, T, η, δ) *with the following properties. Given as input black-box access to $f\colon \{-1,1\}^n \to \{-1,1\}$, a subset $T \subseteq [n]$, and parameters $\eta, \delta > 0$, it runs in time $O(n \cdot \log(1/\delta)/\eta^4)$, makes $O(\log(1/\delta)/\eta^4)$ queries, and with probability at least $1-\delta$ outputs an estimate of $\sum_{i \in T} \hat{f}(i)^2$ that is accurate to within an additive $\pm\eta$.*

Checking Fourier regularity. We recall Lemma 18 of [26], which is an easy consequence of Lemma 3:

▶ **Lemma 4** (Lemma 18 [26]). *There is a procedure* `Check-Fourier-Regular`(f, T, τ, δ) *with the following properties. Given as input black-box access to $f\colon \{-1,1\}^n \to \{-1,1\}$, $T \subseteq [n]$, and $\tau, \delta > 0$, it runs in time $O(n \cdot \log(1/\delta)/\tau^{16})$, makes $O(\log(1/\delta)/\tau^{16})$ queries, and*

- *If $|\hat{f}(i)| \geq \tau$ for some $i \in T$ then it outputs "not regular" with probability $1-\delta$;*
- *If every $i \in T$ has $|\hat{f}(i)| \leq \tau^2/4$ then it outputs "regular" with probability $1-\delta$.*

Estimating the mean. For completeness we recall the following simple fact (which follows from a standard Chernoff bound):

▶ **Fact 5.** *There is a procedure* `Estimate-Mean`(f, ϵ, δ) *with the following properties. Given as input black-box access to* $f\colon \{-1,1\}^n \to \{-1,1\}$ *and* $\epsilon, \delta > 0$*, it makes* $O(\log(1/\delta)/\epsilon^2)$ *queries and with probability at least* $1-\delta$ *it outputs a value* $\tilde{\mu}$ *such that* $|\tilde{\mu} - \mu| \le \epsilon$*, where* $\mu = \mathbf{E}_{\mathbf{x}\in\{-1,1\}^n}[f(\mathbf{x})]$.

The edge tester of [23]. We recall the performance guarantee of the "edge tester" (which works by querying both endpoints of uniform random edges and outputting "non-monotone" if and only if it encounters an anti-monotone edge):

▶ **Theorem 6** ([23]). *There is a procedure* `Edge-Tester`(f, ϵ, δ) *with the following properties: Given black-box access to* $f\colon \{-1,1\}^n \to \{-1,1\}$ *and parameters* $\epsilon, \delta > 0$*, it makes* $O(n \log(1/\delta)/\epsilon)$ *queries and outputs either "monotone" or "non-monotone" such that:*

- *If* f *is monotone then it outputs "monotone" with probability 1;*
- *If* f *is* ϵ*-far from monotone then it outputs "non-monotone" with probability at least* $1-\delta$.

2.5 Useful structural results from prior work

Gaussian distributions and the Berry–Esséen theorem. Recall that the p.d.f. of the standard Gaussian distribution $\mathcal{N}(0,1)$ with mean 0 and variance 1 is given by

$$\phi(x) = \frac{1}{\sqrt{2\pi}} \cdot e^{-x^2/2}.$$

The Berry–Esséen theorem (see e.g., [20]) is a version of the central limit theorem for sums of independent random variables (stating that such a sum converges to a normal distribution) that provides a quantitative error bound. It is useful for analyzing weight-regular LTFs and we recall it below (as well as the standard Hoeffding inequality).

▶ **Theorem 7** (Berry–Esséen). *Let* $\ell(\mathbf{x}) = c_1\mathbf{x}_1 + \cdots + c_n\mathbf{x}_n$ *be a linear form of* n *unbiased, independent random* $\{\pm 1\}$*-valued variables* $\mathbf{x}_i$*. Let* τ *be such that* $|c_i| \le \tau$ *for all* i*, and let* $\sigma = (\sum c_i^2)^{1/2}$*. Write* F *for the c.d.f. of* $\ell(\mathbf{x})/\sigma$*, i.e.,* $F(t) = \Pr[\ell(\mathbf{x})/\sigma \le t]$*. Then for all* $t \in \mathbb{R}$*, we have that* $|F(t) - \Phi(t)| \le \tau/\sigma$*, where* Φ *denotes the c.d.f. of a standard* $\mathcal{N}(0,1)$ *Gaussian random variable.*

▶ **Theorem 8** (Hoeffding's Inequality). *Let* $\mathbf{x}$ *be a random variable drawn uniformly from* $\{-1,1\}^n$*. Let* $w \in \mathbb{R}^d$ *and* $t > 0$*. Then we have*

$$\Pr_{\mathbf{x}}\left[|\mathbf{x}\cdot w| \ge t\right] \le 2\exp\left(-\frac{t^2}{2\|w\|_2^2}\right) \quad \textit{and} \quad \Pr_{\mathbf{x}}\left[\mathbf{x}\cdot w \ge t\right] \le \exp\left(-\frac{t^2}{2\|w\|_2^2}\right).$$

Weight-regularity versus Fourier-regularity for LTFs. An easy argument, using the Berry–Esséen, shows that weight-regularity always implies Fourier-regularity for LTFs:

▶ **Theorem 9** (Theorem 38 of [26]). *Let* $f\colon \{-1,1\}^n \to \{-1,1\}$ *be a* τ*-weight-regular LTF. Then* f *is* $O(\tau)$*-Fourier-regular.*

The converse is not always true; for example, the constant 1 function, which is τ-Fourier-regular for all $\tau > 0$, may be written as $f(x) = \text{sign}(x_1 + 2)$. However, if we additionally impose the condition that f is not too biased towards $+1$ or -1, then a converse holds. Sharpening an earlier result (Theorem 39 of [26]), Dzindzalieta has proved the following:

▶ **Theorem 10** (Theorem 20 of [18]). *Let* $f(x) = \text{sign}(w \cdot x - \theta)$ *be an LTF such that* $|\mathbf{E}_{\mathbf{x}}[f(\mathbf{x})]| \le 1 - \gamma$*. If* f *is* τ*-Fourier-regular, then it is also* $O(\tau/\gamma)$*-weight-regular.*

Making LTFs Fourier-regular by fixing high-influence variables. Finally, we will need the following simple result (Proposition 62 from [26]), which shows that LTFs typically become Fourier-regular when their highest-influence variables are fixed to constants:

▶ **Proposition 11.** *Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be an LTF and let $J \supseteq \{i : |\hat{f}(i)| \geq \beta\}$. Then f_ρ is not (β/η)-Fourier-regular for at most an η-fraction of all $2^{|J|}$ restrictions ρ that fix variables in J.*

3 New structural results about LTFs

Our analysis requires a few new structural results about LTFs. We collect these results in this section; their proofs can be found in the full version.

First we show that, for weight-regular LTFs, the distance to monotonicity corresponds (approximately) to its total amount of squared weights of negative coefficients (under any representation (w, θ)). Lemma 12 below shows that if f is far from monotone then this quantity is large, and Lemma 13 establishes a converse (both for weight-regular LTFs). We note that Lemma 12 is essentially equivalent to a lemma proved in [4].

We introduce some notation. Given an LTF $f\colon \{-1,1\}^n \to \{-1,1\}$ with $f(x) = \mathrm{sign}(w \cdot x - \theta)$, we let $P = P(f)$ and $N = N(f)$ denote the set of non-negative and negative indices, respectively: $P = \{i \in [n]\colon w_i \geq 0\}$ and $N = \{j \in [n]\colon w_j < 0\}$. We let $\mathrm{pos}(f)$ and $\mathrm{neg}(f)$ denote the sum of squared weights of positive and negative coefficients, respectively:

$$\mathrm{pos}(f) = \sum_{i \in P} w_i^2 \quad \text{and} \quad \mathrm{neg}(f) = \sum_{j \in N} w_j^2.$$

Recall that we say f has λ-*significant squared negative weights* if $\mathrm{neg}(f)/(\mathrm{pos}(f)+\mathrm{neg}(f)) \geq \lambda$.

We state Lemma 12 and Lemma 13. Their proofs can be found in the full version.

▶ **Lemma 12.** *Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be an LTF given by $f(x) = \mathrm{sign}(w \cdot x - \theta)$. If f is both ϵ-far from monotone and τ-weight-regular for some $\tau \leq \epsilon/16$, then f must have λ-significant squared negative weights, where $\lambda = \epsilon^2/(16\ln(8/\epsilon))$.*

▶ **Lemma 13.** *Let $f(x) = \mathrm{sign}(\sum_{i\in[n]} w_i \cdot x_i - \theta)$ be (τ, γ, λ)-non-monotone with $\tau \leq \sqrt{\lambda}/16$. Then we have*

$$\mathrm{dist}(f, \textsc{Mono}) \geq \min\left\{\Omega\big(\sqrt{\lambda}\gamma^2\big) - O(\tau), \Omega\left(\frac{\gamma^3}{\ln(8/\gamma)}\right) - O(\tau\gamma)\right\}.$$

Our next goal is to show that for any LTF $f\colon \{-1,1\}^n \to \{-1,1\}$, a random restriction that fixes variables of f that are monotonically non-decreasing has, in expectation, the same distance to monotonicity as the original function f. We state Lemma 14 below, which will be used later in the proof of Lemma 19. Its proof can be found in the full version.

▶ **Lemma 14.** *Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be an LTF and let $S \subseteq [n]$ be a set of variables of f that are monotonically non-decreasing. Then a random restriction $\boldsymbol{\rho}$ that fixes each variable in S independently and uniformly to a random element of $\{-1,1\}$ satisfies*

$$\mathop{\mathbf{E}}_{\boldsymbol{\rho}}\Big[\mathrm{dist}(f_{\boldsymbol{\rho}}, \textsc{Mono})\Big] = \mathrm{dist}(f, \textsc{Mono}).$$

4 Algorithmic tools for LTFs

Our algorithm uses a few simple subroutines that may be viewed as relatively low-level algorithmic tools for working with LTFs. We present these tools in this section; the underlying algorithms and their analysis can be found in the full version.

We start with a subroutine `Find-Hi-Influence-Vars` that finds high-influence variables.

▶ **Lemma 15.** *Suppose that the subroutine* `Find-Hi-Influence-Vars`(f, ρ, τ, δ) *is called on a function* $f\colon \{-1,1\}^n \to \{-1,1\}$*, a restriction* $\rho \in \{-1,1,*\}^n$*, and parameters* $\tau, \delta > 0$*. Then it runs in* $\tilde{O}(\log n \cdot \log(1/\delta)/\tau^{10}) \cdot n$ *time, makes at most* $\tilde{O}(\log n \cdot \log(1/\delta)/\tau^{10})$ *queries, and with probability at least* $1-\delta$ *it outputs a set* $H \subseteq \textsc{stars}(\rho)$ *such that:*

- *If* $|\widehat{f_\rho}(i)| \geq \tau$ *then* $i \in H$*;*
- *If* $|\widehat{f_\rho}(i)| < \tau/2$ *then* $i \notin H$*.*

Given an LTF, the next subroutine `Check-Weight-Positive` checks whether the weight of a variable is positive.

▶ **Lemma 16.** *Suppose that the subroutine* `Check-Weight-Positive`$(f, \rho, i, \tau, \delta)$ *is called on an LTF* $f(x) = \text{sign}(\sum_{i=1}^n w_i x_i - \theta)$*, a restriction* $\rho \in \{-1,1,*\}^n$*,* $i \in \textsc{stars}(\rho)$*, and two parameters* $\tau, \delta > 0$ *such that* $|\hat{f}_\rho(i)| \geq \tau$ *(note that the latter implies that* $w_i \neq 0$*). Then it runs in* $O(\log(1/\delta)/\tau) \cdot n$ *time, makes* $O(\log(1/\delta)/\tau)$ *queries, and:*

- *If it does not output "fail", which happens with probability at most* δ*;*
- *It outputs "positive" if* $w_i > 0$*, and it outputs "negative" if* $w_i < 0$*.*

5 Detailed description of the algorithm

We present our algorithm and its analysis in this section.

5.1 The algorithm

Our main testing algorithm, `Mono-Test-LTF`, is presented in Figure 1. Its main components are two procedures called `Regularize-and-Balance` and `Main-Procedure`, described and analyzed in Sections 5.2 and 5.3. As will become clear later, `Mono-Test-LTF` is one-sided, i.e., it always outputs "monotone" when the input function f is monotone (because it only outputs "non-monotone" when an anti-monotone edge is found, via `Check-Weight-Positive` or `Edge-Tester`). Thus, our analysis of correctness below focuses on the case when f is an LTF that is ϵ-far from monotone, and shows that in this case `Mono-Test-LTF` outputs "non-monotone" with probability at least 2/3.

5.2 Key properties of procedure `Regularize-and-Balance`

Let $f\colon \{-1,1\}^n \to \{-1,1\}$ be an LTF, given by $f(x) = \text{sign}(w \cdot x - \theta)$. Assume that f is ϵ-far from monotone. The goal of the procedure `Regularize-and-Balance`(f, ϵ) is to return a restriction $\rho \in \{-1,1,*\}^{[n]}$ such that f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF (with respect to (w, θ)), where

$$\lambda = \frac{\epsilon^2}{36\ln(12/\epsilon)} \quad \text{and} \quad \tau = \frac{\lambda\epsilon}{\log^2 n}. \tag{2}$$

Here is some intuition that may be helpful in understanding `Regularize-and-Balance`. If the procedure halts and outputs "monotone" in Step 2, this signals that the (low-probability) failure event of `Find-Hi-Influence-Variables` has taken place (since it has

```
Algorithm Mono-Test-LTF($f, \epsilon$)
Input: Oracle access to an LTF $f\colon \{-1,1\}^n \to \{-1,1\}$ and a parameter $\epsilon > 0$.
Output: Returns "monotone" or "non-monotone."
1. Call Regularize-and-Balance($f, \epsilon$). If it returns a restriction $\rho \in \{-1,1,*\}^{[n]}$ then
   continue to Step 2; if it returns "non-monotone," halt and output "non-monotone;"
   if it returns "monotone," halt and output "monotone."
2. Call Main-Procedure($f, \rho, \epsilon$). If it returns "non-monotone," halt and output "non-
   monotone;" if it returns "monotone," halt and output "monotone."
```

Figure 1 Main algorithm `Mono-Test-LTF`. If f is monotone it outputs "monotone" with probability 1; if f is ϵ-far from monotone, it outputs "non-monotone" with probability $\geq 2/3$.

spuriously identified more variables as having high influence than is possible given Parseval's identity; see Lemma 15). The procedure halts and outputs "non-monotone" in Step 3 only if `Check-Weight-Positive` has unambiguously found an anti-monotone edge. If the procedure outputs "monotone" in Step 3, this signals the (low-probability) event that `Check-Weight-Positive` failed to identify some index $i \in H$ (which was supposed to have high influence) as either having $w_i > 0$ or $w_i < 0$. Finally if it outputs "monotone" in Step 4, this signals that f appears to be close to monotone.[7]

It is clear that `Regularize-and-Balance` is one-sided.

▶ **Fact 17.** *`Regularize-and-Balance`(f, ϵ) never returns "non-monotone" if f is monotone.*

We also have the following upper bound for the number of queries it uses (which can be straight forwardly verified by tracing through procedure calls and parameter settings):

▶ **Fact 18.** *The number of queries used by `Regularize-and-Balance`(f, ϵ) is $\tilde{O}(\log^{41} n/\epsilon^{90})$.*

We prove the main property of the procedure `Regularize-and-Balance` in Appendix A.

▶ **Lemma 19.** *If $f(x) = \text{sign}(w \cdot x - \theta)$ is ϵ-far from monotone, then with probability at least $9/10$, `Regularize-and-Balance`(f, ϵ) returns either "non-monotone," or a restriction ρ such that f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF with respect to (w, θ).*

5.3 Key properties of `Main-Procedure`

`Main-Procedure` is presented in Figure 3. Given Lemma 19 we may assume that the input (f, ρ, ϵ) satisfies that f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF (see the choices of τ and λ in (2)).

We prove the following main lemma in this section.

▶ **Lemma 20.** *`Main-Procedure`(f, ρ, ϵ) never returns "non-monotone" when f is monotone. When f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF, it returns "non-monotone" with probability at least $81/100$.*

The procedure only returns "non-monotone" when it finds an anti-monotone edge in the subroutine `Check-Weight-Positive`. Hence we may focus on the case when f_ρ is $(\tau, \epsilon, \lambda)$-non-monotone. For this purpose, we analyze the three steps 2(a), 2(b), 2(c) of each while loop of `Main-Procedure`, and prove the following lemma.

[7] This will become clear later in the proof of Lemma 19 where we show that Step 4 fails with low probability when f is far from monotone.

```
Procedure Regularize-and-Balance($f, \epsilon$)
Input: Parameter $\epsilon > 0$ and black-box oracle access to an LTF $f\colon \{-1,1\}^n \to \{-1,1\}$
of the form $f(x) = \mathrm{sign}(w \cdot x - \theta)$, with unknown weights $w$ and threshold $\theta$.
Output: Either "non-monotone," "monotone," or a restriction $\rho \in \{-1,1,*\}^{[n]}$.

1. Let $C_{RB} > 0$ be a large enough constant; let $\tau'$ and $\delta$ be the following parameters:

       $\tau' = \tau^2\epsilon^3/C_{RB}$   and   $\delta = \tau'^2/C_{RB}$.

2. Call Find-Hi-Influence-Vars($f, (*)^n, \tau', \delta$) and let $H$ be the set it returns.
   If $|H| > 4/\tau'^2$, halt and output "monotone."
3. For each $i \in H$, call Check-Weight-Positive($f, (*)^n, i, \tau'/2, \delta$). If any call returns
   "negative," halt and output "non-monotone;" if any call returns "fail," halt and output
   "monotone;" otherwise (when all calls return "positive") continue to Step 4.
4. Repeat $C_{RB}/\epsilon$ times:
      Draw a restriction $\rho$, which has support $H$ and is obtained by selecting a random
      assignment from $\{-1,1\}^H$. Call

          Check-Fourier-Regular($f_\rho, [n] \setminus H, \sqrt{12\tau'/\epsilon}, \delta/2$)

      and Estimate-Mean($f_\rho, \epsilon/6, \delta/2$).
   Halt and output the first $\rho$ where Check-Fourier-Regular outputs "regular" and
   Estimate-Mean returns a number of absolute value $\le 1 - 7\epsilon/6$. If the procedure fails
   to find such a restriction $\rho$, halt and output "monotone."
```

Figure 2 Procedure Regularize-and-Balance. Our analysis (Lemma 19) focuses on the case when f is ϵ-far from monotone.

▶ **Lemma 21.** *Let $t \le 4\log n$, and suppose that at the beginning of the $(t+1)$th loop of* Main-Procedure*, $f_{\rho^{(t)}}$ is $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone. Then with probability at least $1 - 1/(40\log n)$, it either returns "non-monotone" within this loop or obtains a set $A_t \subseteq [n] \setminus \mathrm{supp}(\rho^{(t)})$ and a restriction $\rho^{(t+1)}$ extending $\rho^{(t)}$ at the end of this loop such that*

1. $|A_t| \ge |\textsc{stars}(\rho^{(t)})|/4$;
2. $\mathrm{supp}(\rho^{(t)}) \cup A_t \subseteq \mathrm{supp}(\rho^{(t+1)})$*; and*
3. $f_{\rho^{(t+1)}}$ *is a $(\tau, \epsilon, \lambda(1 - (t+1)/(8\log n)))$-non-monotone LTF.*

We use Lemma 21 to prove Lemma 20 in Appendix B.1.

5.3.1 Proof of Lemma 21

The proof of Lemma 21 consists of three lemmas, one for each steps 2(a), 2(b) and 2(c). Below we assume that the condition of Lemma 21 holds at the beginning of the $(t+1)^{\text{th}}$ loop, for some $t \le 4\log n$. We introduce the following notation for convenience. We let $I = \textsc{stars}(\rho^{(t)})$, with $m = |I|$. Given the random subset A_t of I found in Step 2(a), we let $B_t = I \setminus A_t$. Also note that $m \ge 1/\tau^2$.

We start with the lemma for Step 2(a), which states that with high probability, A_t is large and splits the weights (both positive and negative) in I evenly. We present the proof in Appendix B.2.

```
Procedure Main-Procedure($f, \rho, \epsilon$)
Input: Parameter $\epsilon > 0$, oracle access to an LTF $f\colon \{-1,1\}^n \to \{-1,1\}$ of the form
$f(x) = \mathrm{sign}(w \cdot x - \theta)$ with unknown weights $w$ and threshold $\theta$, and a restriction $\rho$.
Output: Either "non-monotone" or "monotone."
1. Set $t = 0$ and $\rho^{(0)} = \rho$.
2. While $|\textsc{stars}(\rho^{(t)})| \geq 1/\tau^2$, repeat the following steps:
    a. Construct a subset $A_t \subseteq \textsc{stars}(\rho^{(t)})$ by independently putting each index $i \in$
       $\textsc{stars}(\rho^{(t)})$ into $A_t$ with probability $1/2$.
    b. Call Find-Balanced-Restriction($f, \rho^{(t)}, A_t, \epsilon$). If it returns "monotone"
       then halt and return "monotone;" otherwise, it returns a restriction $\rho'$ with
       $\mathrm{supp}(\rho') = \mathrm{supp}(\rho^{(t)}) \cup A_t$.
    c. Call Maintain-Regular-and-Balance($f, \rho', \epsilon$). If it returns "non-monotone"
       then halt and output "non-monotone;" if it returns "monotone" then halt and
       output "monotone;" otherwise, it returns a restriction $\eta$ and we set $\rho^{(t+1)}$ to $\rho'\eta$.
    d. Increment $t$ by 1. If $t > 4\log n$, halt and output "monotone;" otherwise proceed
       to the next iteration of step (a) of the loop.
3. Let $\epsilon' = \epsilon^3/(C\log(1/\epsilon))$ for some large constant $C$; run Edge-Tester($f_{\rho^{(t)}}, \epsilon', 1/10$)
   and output what it outputs (either "monotone" or "non-monotone").
```

Figure 3 Procedure Main-Procedure. Our analysis in Section 5.3 focuses on the case when f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF.

```
Subroutine Find-Balanced-Restriction($f, \rho^{(t)}, A_t, \epsilon$)
Input: Access to $f\colon \{-1,1\}^n \to \{-1,1\}$, restriction $\rho^{(t)}$, $A_t \subseteq \textsc{stars}(\rho^{(t)})$, and $\epsilon > 0$.
Output: "monotone" or a $\rho'$ with $\mathrm{supp}(\rho') = \mathrm{supp}(\rho^{(t)}) \cup A_t$ that extends $\rho^{(t)}$.
    Repeat $C_{BR} \cdot \log n/\epsilon^3$ times for some large enough constant $C_{BR}$:
        Draw a $\rho^*$, which has support $A_t$ and is obtained by selecting a random assignment
        from $\{-1,1\}^{A_t}$, and let $\rho' = \rho^{(t)}\rho^*$. Call Estimate-Mean($f_{\rho'}, 0.01, \delta$), where
        $\delta = \epsilon^3/(200 C_{BR} \log^2 n)$. If it returns a number of absolute value at most 0.03,
        halt and output $\rho'$.
    Otherwise, output "monotone."
```

Figure 4 Subroutine Find-Balanced-Restriction. We are interested in the case when $f_{\rho^{(t)}}$ is a $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone LTF, and A_t satisfies the conditions of Lemma 23.

▶ **Lemma 22.** *Assume that $f_{\rho^{(t)}}$ is a $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone LTF. With probability at least $1 - \exp(-\Omega(\log^2 n))$, A_t and B_t satisfy $|A_t| \geq m/4$,*

$$\frac{1}{2} - \frac{1}{32\log n} \leq \frac{\sum_{i \in A_t} w_i^2}{\sum_{i \in I} w_i^2} \leq \frac{1}{2} + \frac{1}{32\log n} \quad \textit{and} \quad \frac{\sum_{i \in B_t : w_i < 0} w_i^2}{\sum_{i \in B_t} w_i^2} \geq \lambda\left(1 - \frac{t+1}{8\log n}\right). \tag{3}$$

We give Find-Balanced-Restriction in Figure 4 and show the following lemma for Step 2(b). (The Find-Balanced-Restriction subroutine is similar to Algorithm 1 of [28], and Lemma 23 and its proof (presented in Appendix B.3) are reminiscent of Lemma 7 of [28]; however, because of some technical differences we cannot directly apply those results, so we give a self-contained presentation here.)

```
Subroutine Maintain-Regular-and-Balanced($f, \rho', \epsilon$)
Input: Oracle access to $f\colon \{-1,1\}^n \to \{-1,1\}$, restriction $\rho'$, parameter $\epsilon > 0$.
Output: "non-monotone," "monotone," or an $\eta$ with $\mathrm{supp}(\eta) \subseteq B_t$ extending $\rho'$.

1. Let $C_M > 0$ be a large enough constant; let $\tau', \delta$ and $\tau^*$ be the following parameters:
       $\tau' = (\tau\epsilon/C_M)^2 \cdot \sqrt{\lambda}, \quad \delta = \tau'^2/(C_M \log n) \quad \text{and} \quad \tau^* = \tau'/\sqrt{\lambda}.$
2. Call Find-Hi-Influence-Vars($f, \rho', \tau', \delta$) and let $H$ be the set that it returns.
   If $|H| > 4/\tau'^2$, halt and return "monotone."
3. For each $i \in H$, call Check-Weight-Positive($f, \rho', i, \tau'/2, \delta$). If any call returns
   "negative" then halt and output "non-monotone;" if any call returns "fail" then halt
   and output "monotone;" otherwise (every call returns "positive") continue to Step 4.
4. Repeat $C_M \log n/\sqrt{\lambda}$ times:
       Draw a restriction $\eta$ with support $H$, by selecting a random assignment from
       $\{-1,1\}^H$. Call Check-Fourier-Regular($f_{\rho'\eta}, [n] \setminus \mathrm{supp}(\rho'\eta), \sqrt{C_M \tau^*}, \delta/2$) and
       Estimate-Mean($f_{\rho'\eta}, \epsilon/6, \delta/2$).
   Halt and output the first restriction $\eta$ where Check-Fourier-Regular outputs "regular"
   and Estimate-Mean returns a number of absolute value $\le 1 - 7\epsilon/6$. If the
   procedure fails to find such a restriction $\eta$, halt and output "monotone."
```

Figure 5 Subroutine `Maintain-Regular-and-Balanced`. Lemma 24 assumes that $f_{\rho^{(t)}}$ is an $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone LTF, $|A_t| \ge m/4$ and (3), and $f_{\rho'}$ is 0.96-balanced.

▶ **Lemma 23.** *Assume that $f_{\rho^{(t)}}$ is a $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone LTF, and sets A_t and B_t satisfy $|A_t| \ge m/4$ and (3). With probability at least $1/(100\log n)$, the subroutine `Find-Balanced-Restriction` outputs a restriction ρ' with $\mathrm{supp}(\rho') = \mathrm{supp}(\rho^{(t)}) \cup A_t$ such that ρ' extends $\rho^{(t)}$ and $f_{\rho'}$ is 0.96-balanced.*

For Step 2(c) of `Main-Procedure`, the subroutine `Maintain-Regular-and-Balanced` is given in Figure 5. It is very similar to `Regularize-and-Balance` except the number of rounds in Step 4 and the choice of parameters τ' and δ. We leave the proof of the following lemma to the full version. Lemma 21 follows directly from Lemmas 22, 23, and 24.

▶ **Lemma 24.** *Suppose that $f_{\rho^{(t)}}$ is a $(\tau, \epsilon, \lambda(1 - t/(8\log n)))$-non-monotone LTF, sets A_t and B_t satisfy $|A_t| \ge m/4$ and (3), and $f_{\rho'}$ is 0.96-balanced. Then with probability at least $1 - 1/(100\log n)$, `Maintain-Regular-and-Balance` returns either "non-monotone," or a restriction η with $\mathrm{supp}(\eta) \subseteq B_t$ such that $f_{\rho^{(t+1)}}$, where $\rho^{(t+1)} = \rho'\eta$, is $(\tau, \epsilon, \lambda(1 - (t+1)/(8\log n)))$-non-monotone.*

5.4 Final analysis of the algorithm

We conclude by stating the correctness and query complexity of the algorithm. The proofs of the following two theorems appear in Appendix C.

▶ **Theorem 25.** *The algorithm `Mono-Test-LTF`(f, ϵ) correctly tests whether a given LTF is monotone or ϵ-far from monotone.*

▶ **Theorem 26.** *The algorithm `Mono-Test-LTF`(f, ϵ) makes $\tilde{O}(\log^{42} n/\epsilon^{90})$ queries.*

Theorem 1 follows as an immediate consequence of Theorems 25 and 26.

References

1 N. Ailon, B. Chazelle, S. Comandur, and D. Liu. Estimating the distance to a monotone function. *Random Structures and Algorithms*, 31(3):371–383, 2007.

2 Roksana Baleshzar, Deeparnab Chakrabarty, Ramesh Krishnan S. Pallavoor, Sofya Raskhodnikova, and C. Seshadhri. Optimal unateness testers for real-values functions: Adaptivity helps. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP '2017)*, 2017.

3 T. Batu, R. Kumar, and R. Rubinfeld. Sublinear algorithms for testing monotone and unimodal distributions. In *Proceedings of the 36th ACM Symposium on Theory of Computing*, pages 381–390, 2004.

4 A. Belovs and E. Blais. A polynomial lower bound for testing monotonicity. In *Proceedings of the 48th ACM Symposium on Theory of Computing*, 2016.

5 Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev. L_p-testing. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 164–173, 2014.

6 E. Blais, J. Brody, and K. Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012.

7 E. Blais, S. Raskhodnikova, and G. Yaroslavtsev. Lower bounds for testing properties of functions on hypergrid domains. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:36, 2013.

8 J. Briët, S. Chakraborty, D. García-Soriano, and A. Matsliah. Monotonicity testing and shortest-path routing on the cube. *Combinatorica*, 32(1):35–53, 2012.

9 C. Canonne, D. Ron, and R. Servedio. Testing probability distributions using conditional samples. *SIAM Journal on Comput.*, 44(3):540–616, 2015.

10 D. Chakrabarty and C. Seshadhri. A $o(n)$ monotonicity tester for boolean functions over the hypercube. In *Proceedings of the 45th ACM Symposium on Theory of Computing*, pages 411–418, 2013.

11 D. Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *Proceedings of the 45th ACM Symposium on Theory of Computing*, pages 419–428, 2013.

12 D. Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. *Theory of Computing*, 10(17):453–464, 2014.

13 X. Chen, A. De, R. A. Servedio, and L.-Y. Tan. Boolean function monotonicity testing requires (almost) $n^{1/2}$ non-adaptive queries. In *Proceedings of the 47th ACM Symposium on Theory of Computing*, pages 519–528, 2015.

14 X. Chen, R. A. Servedio, and L.-Y. Tan. New algorithms and lower bounds for monotonicity testing. In *Proceedings of the IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 286–295, 2014.

15 Xi Chen, Rocco A. Servedio, Li-Yang Tan, Erik Waingarten, and Jinyu Xie. Settling the query complexity of non-adaptive junta testing. In *Proceedings of the 32nd Conference on Computational Complexity (CCC '2017)*, 2017.

16 Xi Chen, Erik Waingarten, and Jinyu Xie. Beyond talagrand functions: new lower bounds for testing monotonicity and unateness. In *Proceedings of the 49th ACM Symposium on the Theory of Computing (STOC '2017)*, 2017.

17 Y. Dodis, O. Goldreich, E. Lehman, S. Raskhodnikova, D. Ron, and A. Samorodnitsky. Improved testing algorithms for monotonocity. In *Proceedings of the 3rd International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 97–108, 1999.

18 D. Dzindzalieta. Tight Bernoulli tail probability bounds. Technical Report Doctoral Dissertation, Physical Sciences, Mathematics (01 P), Vilnius University, 2014.

19 F. Ergün, S. Kannan, S. R. Kumar, R. Rubinfeld, and M. Vishwanthan. Spot-checkers. *Journal of Computer and System Sciences*, 60:717–751, 2000.

20 W. Feller. *An introduction to probability theory and its applications.* John Wiley & Sons, 1968.

21 E. Fischer. On the strength of comparisons in property testing. *Information and Computation*, 189(1):107–116, 2004.

22 E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, R. Rubinfeld, and A. Samorodnitsky. Monotonicity testing over general poset domains. In *Proceedings of the 34th Annual ACM Symposium on the Theory of Computing*, pages 474–483, 2002.

23 O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samordinsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

24 S. Halevy and E. Kushilevitz. Testing monotonicity over graph products. *Random Structures and Algorithms*, 33(1):44–67, 2008.

25 S. Khot, D. Minzer, and M. Safra. On monotonicity testing and boolean isoperimetric type theorems. In *Proceedings of the 56th Annual Symposium on Foundations of Computer Science*, pages 52–58, 2015.

26 K. Matulef, R. O'Donnell, R. Rubinfeld, and R. Servedio. Testing halfspaces. *SIAM Journal on Comput.*, 39(5):2004–2047, 2010.

27 D. Ron, R. Rubinfeld, M. Safra, A. Samorodnitsky, and O. Weinstein. Approximating the influence of monotone Boolean functions in $O(\sqrt{n})$ query complexity. *ACM Transactions on Computation Theory*, 4(4):1–12, 2012.

28 D. Ron and R. A. Servedio. Exponentially improved algorithms and lower bounds for testing signed majorities. *Algorithmica*, 72(2):400–429, 2015.

29 R. Rubinfeld and R. A. Servedio. Testing monotone high-dimensional distributions. *Random Structures and Algorithms*, 34(1):24–44, 2009. doi:10.1002/rsa.20247.

30 R. A. Servedio, L.-Y. Tan, and J. Wright. Adaptivity helps for testing juntas. In *Proceedings of the 30th IEEE Conference on Computational Complexity*, pages 264–279, 2015.

A Proof of Lemma 19

Proof. Using Lemma 15, with probability $1-\delta$, `Find-Hi-Influence-Vars` in Step 2 returns a set $H \subseteq [n]$ of indices that satisfies the following property:

$$\text{If } |\hat{f}(i)| \geq \tau' \text{ then } i \in H; \text{ If } |\hat{f}(i)| < \tau'/2 \text{ then } i \notin H. \tag{4}$$

When this happens, we have by Parseval $|H| \leq 4/\tau'^2$, and the procedure continues to Step 3.

We consider two subevents: E_0': H satisfies (4) but contains an elements i with $w_i < 0$; and E_0: H satisfies (4) and every $i \in H$ has $w_i > 0$. We have $\Pr[E_0'] + \Pr[E_0] \geq 1-\delta$ as discussed above. Below we show that the procedure returns "non-monotone" with high probability, conditioning on E_0', and it returns a restriction with the desired property with high probability, conditioning on E_0. By the end we combine the two cases to conclude that

$$\begin{aligned}&\Pr[E_0'] \cdot \Pr\big[\text{the procedure returns "non-monotone"} \mid E_0'\big]\\ &\qquad + \Pr[E_0] \cdot \Pr\big[\text{it returns } \rho \text{ such that } f_\rho \text{ is } (\tau,\epsilon,\lambda)\text{-non-monotone} \mid E_0\big] \geq 9/10.\end{aligned}$$

We first address the (easier) case of E_0'. Assume $i \in H$ satisfies $w_i < 0$. From (4), $|\hat{f}(i)| \geq \tau'/2$ and thus, `Check-Weight-Positive`$(f, (*)^n, i, \tau'/2, \delta)$ in Step 3 returns "negative" with probability $1-\delta$, and the procedure returns "non-monotone" with probability $1-\delta$, conditioning on E_0'.

Next we address the (harder) case of E_0. First we use E_1 to denote the event that every call to `Check-Weight-Positive` in Step 3 returns the correct answer, i.e., it returns "positive" for every $i \in H$. By a union bound we have $\Pr[E_1 \mid E_0] \geq 1 - 4\delta/\tau'^2$.

Assuming that E_1 happens, the procedure proceeds to Step 4 and we use E_2 to denote the event that `Check-Fourier-Regular` and `Estimate-Mean` return the correct answer, i.e.:

1. `Check-Fourier-Regular` outputs "not regular" if $|\hat{f}_\rho(i)| \geq \sqrt{12\tau'/\epsilon}$ for some $i \in [n] \setminus H$, and outputs "regular" if $|\hat{f}_\rho(i)| \leq 3\tau'/\epsilon$ for all $i \in [n] \setminus H$, for every ρ in Step 4, and
2. `Estimate-Mean` returns a number a with $|a - \mathbf{E}[f_\rho]| \leq \epsilon/6$, for every ρ in Step 4.

We also write E_3 to denote the event that one of the restrictions ρ drawn in Step 4 satisfies that f_ρ is both $(2\epsilon/3)$-far from monotone and $(3\tau'/\epsilon)$-Fourier-regular. By a union bound, we have that $\Pr[E_2 \mid E_0 \wedge E_1] \geq 1 - C_{RB}\delta/\epsilon$.

In the rest of the proof we show that

1. $\Pr[E_3 \mid E_0 \wedge E_1] \geq 99/100$ and
2. Given E_0, E_1, E_2 and E_3, the procedure always returns a restriction ρ such that f_ρ is $(\tau, \epsilon, \lambda)$-non-monotone.

Together we have that it returns such a ρ with probability at least (conditioning on E_0)

$$(1 - 4\delta/\tau'^2) \cdot (1 - C_{RB}\delta/\epsilon - 1/100).$$

Summarizing the two cases of E_0' and E_0 we have that `Regularize-and-Balance` returns either "non-monotone" or a ρ such that f_ρ is $(\tau, \epsilon, \lambda)$-non-monotone with probability at least

$$\Pr[E_0'] \cdot (1 - \delta) + \Pr[E_0] \cdot (1 - 4\delta/\tau'^2) \cdot (1 - C_{RB}\delta/\epsilon - 1/100) > 9/10,$$

using $\Pr[E_0'] + \Pr[E_0] \geq 1 - \delta$ and our choice of δ (by letting C_{RB} be large enough).

We use the following claim to show that $\Pr[E_3 \mid E_0 \wedge E_1] \geq 99/100$.

▶ **Claim 27.** *A random restriction ρ over H satisfies that f_ρ is both $(2\epsilon/3)$-far from monotone and $(3\tau'/\epsilon)$-Fourier-regular with probability at least $\epsilon/3$.*

Proof. For each of the two properties, we have

1. Proposition 11: with probability at least $1 - (\epsilon/3)$, f_ρ is $(3\tau'/\epsilon)$-Fourier-regular.
2. Lemma 14: with probability at least $2\epsilon/3$, f_ρ is $(2\epsilon/3)$-far from monotone. To see this, let c be the probability of f_ρ being $(2\epsilon/3)$-far from monotone. Then $c \geq 2\epsilon/3$ follows from

$$(1 - c) \cdot (2\epsilon/3) + c \cdot (1/2) \geq \epsilon,$$

where we used the fact that distance to monotonicity is always at most 1/2.

The claim then follows from a union bound. ◀

By choosing C_{RB} to be a large enough constant, we have $\Pr[E_3 \mid E_0 \wedge E_1] \geq 99/100$.

Finally we show that conditioning on all four events E_0, E_1, E_2, E_3 the procedure always returns a restriction ρ such that f_ρ is a $(\tau, \epsilon, \lambda)$-non-monotone LTF. We do this in two steps:

1. First, given E_3, one of the restrictions ρ drawn in Step 4 is both $(2\epsilon/3)$-far from monotone and $(3\tau'/\epsilon)$-Fourier-regular. Given E_2, ρ must pass both tests, i.e., `Check-Fourier-Regular` outputs "regular" and `Estimate-Mean` returns a number of absolute value at most $1 - 7\epsilon/6$ in Step 4. The former is trivial; to see the latter, note that being $(2\epsilon/3)$-far from monotone implies that $|\mathbf{E}[f_\rho]| \leq 1 - 4\epsilon/3$ and therefore, the number returned by `Estimate-Mean` is at most $1 - 7\epsilon/6$, given E_2.

2. Second, we show that if a restriction ρ passes both tests in Step 4 of the procedure, then f_ρ must be $(\tau, \epsilon, \lambda)$-non-monotone. One can think of this as a soundness property, saying that if the procedure halts and returns some ρ, that it returns a correct one. To see this, note that by E_2, f_ρ is both $\sqrt{12\tau'/\epsilon}$-Fourier regular and ϵ-balanced. By Theorem 10, f_ρ is $O(\sqrt{\tau'/\epsilon^3})$-weight-regular, and τ-weight-regular by letting C_{RB} be large enough. It also follows from Lemma 12 that f_ρ has λ-significant squared negative weights.

This finishes the proof of the lemma. ◀

B Proofs of Lemma 20, Lemma 22, and Lemma 23

B.1 Proof of the Second Part of Lemma 20 using Lemma 21

Proof. We consider the event E where the conclusion of Lemma 21 holds for every iteration of the while loop of `Main-Procedure`. As the condition of Lemma 21 holds for the first loop ($t = 0$) and there are at most $4 \log n$ many loops, this happens with probability at least 9/10. Since E implies $|A_t| \geq |\textsc{stars}(\rho^{(t)})|/4$, we can also assume that the procedure never halts and outputs “monotone” due to line 2(d).

Given E, `Main-Procedure` either returns “non-monotone” as desired or reaches line 3. Furthermore, if it reaches line 3, $f_{\rho^{(t)}}$ must be $(\tau, \epsilon, \lambda/2)$-non-monotone by Lemma 21 and have at most $1/\tau^2$ variables. It follows from Lemma 13 that $f_{\rho^{(t)}}$ is ϵ'-far from monotone, where $\epsilon' = \epsilon^3/(C\log(1/\epsilon))$ for some large enough constant C. Finally, by Theorem 6, `Edge-Tester` outputs “non-monotone” (by finding an anti-monotone edge) with probability at least 9/10 and the proof is complete. ◀

B.2 Proof of Lemma 22

Proof. We consider the three events separately and then apply a union bound.

First by Chernoff bound, $|A_t| \geq m/4$ holds with probability at least $1 - e^{-\Omega(m)}$.

Next for the first inequality in (3), assume without loss of generality that $\sum_{i\in I} w_i^2 = 1$ (as $f_{\rho^{(t)}}$ cannot be all-1 or all-(-1)). By Hoeffding bound the probability that it does not hold is at most

$$2\exp\left(-\Omega\left(\frac{1/\log^2 n}{\sum_{i\in I} w_i^4}\right)\right).$$

Since $f_{\rho^{(t)}}$ is τ-weight-regular (over I), we have that $|w_i| \leq \tau$ for all $i \in I$ and thus,

$$\sum_{i\in I} w_i^4 \leq \tau^2 \cdot \sum_{i\in I} w_i^2 = \tau^2.$$

As a result, the second inequality holds with probability at least $1 - \exp(-\Omega(1/(\tau^2 \log^2 n)))$.

For the last inequality, note that $\sum_{i\in I:w_i<0} w_i^2 \geq \lambda(1-t/(8\log n))$. Similarly by Hoeffding,

$$\Pr\left[\sum_{i\in B_t:w_i<0} w_i^2 < \left(\frac{\lambda}{2}\right)\left(1-\frac{t+0.5}{8\log n}\right)\right] \leq \exp\left(-\Omega\left(\frac{\lambda^2/\log^2 n}{\sum_{i\in I:w_i<0} w_i^4}\right)\right) \leq \exp\left(-\Omega\left(\log^2 n\right)\right).$$

Combining the above with the analysis of the first inequality in (3), the last inequality holds with probability at least $1 - \exp(-\Omega(\log^2 n))$. The lemma follows from a union bound. ◀

B.3 Proof of Lemma 23

Proof. For convenience we use f' to denote $f_{\rho^{(t)}}$, w' to denote the weight vector w but restricted on I, and θ' to denote the new threshold, i.e.,

$$\theta' = \theta - \sum_{i\in\text{supp}(\rho^{(t)})} \rho^{(t)}(i)\cdot w_i.$$

Without loss of generality we assume that $\sum_{i\in I} w_i'^2 = 1$. We may additionally assume that $\theta' \geq 0$. This assumption is without loss of generality, because 1) if ρ' is a 0.96-balanced restriction when $-\theta' \geq 0$, then $-\rho'$ is a 0.96-balanced restriction for $\theta' \leq 0$, and 2) `Find-Balanced-Restriction` will test the only take into account the absolute value of the output of `Estimate-Mean`. Let

$$\alpha = \sum_{i\in A_t} w_i'^2 \quad \text{and} \quad \beta = \sum_{i\in B_t} w_i'^2.$$

We use $a = b \pm c$ to denote the inequalities $b - c \leq a \leq b + c$. Then from (3) we have that $\alpha, \beta = 1/2 \pm O(1/\log n)$. By assumption, f' is τ-weight-regular and ϵ-balanced.

For the analysis we define two events E_1 and E_2. Here E_1 denotes the event that every call to `Estimate-Mean` returns a number a such that $|a - \mathbf{E}[f_{\rho'}]| \leq 0.01$. By a union bound, this happens with probability $1 - 1/(200\log n)$. Let E_2 be the event that one of the restrictions ρ^* drawn has f'_{ρ^*} being 0.98-balanced. When E_1 and E_2 both occur, the subroutine outputs a restriction ρ' such that $f_{\rho'}$ is 0.96-balanced. In the rest of the proof we show that event E_2 happens with high probability.

To analyze the probability of f'_{ρ^*} being 0.98-balanced, we use $\mathbf{x}_i$ to denote an independent and unbiased random $\{-1,1\}$-variable for each $i \in I$, and let

$$\mathbf{x}_A = \sum_{i\in A_t} \mathbf{x}_i \cdot w_i', \quad \mathbf{x}_B = \sum_{i\in B_t} \mathbf{x}_i \cdot w_i' \quad \text{and} \quad \mathbf{x} = \mathbf{x}_A + \mathbf{x}_B.$$

By Hoeffding bound and the assumption that f' is ϵ-balanced, we have

$$2\epsilon = \Pr[\mathbf{x} \geq \theta'] \leq \exp(-\theta'^2/2). \tag{5}$$

Using Berry–Esséen $\mathbf{x}_A + \mathbf{x}_B$ is $O(\tau)$-close to a standard $\mathcal{N}(0,1)$ Gaussian random variable, denoted by $\mathcal{G}$, $\mathbf{x}_A$ is $O(\tau)$-close to $\sqrt{\alpha}\mathcal{G}$, and $\mathbf{x}_B$ is $O(\tau)$-close to $\sqrt{\beta}\mathcal{G}$.

Let $\theta^* > 0$ be the threshold such that $\Pr[|\sqrt{\beta}\mathcal{G}| \leq \theta^*] = 0.01$. Then

$$\Pr\left[f'_{\rho^*} \text{ is 0.98-balanced}\right] \geq \Pr\left[\mathbf{x}_A \in [\theta' - \theta^*, \theta' + \theta^*]\right].$$

This is because, for any number $x_A \in [\theta' - \theta^*, \theta' + \theta^*]$, we have

$$0.495 - O(\tau) \leq \Pr\left[\mathbf{x}_B \geq \theta' - x_A\right] = \Pr\left[\sqrt{\beta}\mathcal{G} \geq \theta' - x_A\right] \pm O(\tau) \leq 0.505 + O(\tau),$$

in which case the function f'_{ρ^*} is $0.99 - O(\tau) = 0.98$-balanced. To bound $\Pr\left[\mathbf{x}_A \in [\theta' - \theta^*, \theta' + \theta^*]\right]$, we note that $\theta' \geq 0$ (by assumption) and $\theta^* = \Omega(1)$ (by our choice of θ^* and $\beta > 1/3$). As a result,

$$\Pr\left[\mathbf{x}_A \in [\theta' - \theta^*, \theta']\right] \geq \Pr\left[\sqrt{\alpha}\mathcal{G} \in [\theta' - \theta^*, \theta']\right] - O(\tau) = \Omega(1)\cdot\Omega(\epsilon^3) - O(\tau) = \Omega(\epsilon^3),$$

where we used $\alpha > 1/3$ by (3), $\tau = o(\epsilon^3)$, and $\exp(-\theta'^2/2) = \Omega(\epsilon)$ from (5) to obtain

$$\min\Big(\exp\left(-(\theta^*)^2/(2\alpha)\right), \exp\left(-\theta'^2/(2\alpha)\right)\Big) = \Omega(\epsilon^3).$$

As a result, a random restriction ρ^* is 0.98-balanced with probability at least $\Omega(\epsilon^3)$. Thus with probability $1-1/n$ (by choosing a large enough constant C_{BR}), `Find-Balanced-Restriction` gets such a restriction that would pass the `Estimate-Mean` test. By a union bound on E_1 and E_2, `Find-Balanced-Restriction` returns a 0.96-balanced ρ' with probability at least $1 - 1/(200 \log n) - 1/n > 1 - 1/(100 \log n)$. This finishes the proof of the lemma. ◀

C Proofs of the Final Analysis

Proof of Theorem 25. The algorithm is one-sided because it outputs "non-monotone" only when an anti-monotone edge is found. The only interesting case is when the input LTF f is ϵ-far from monotone. Combining Lemmas 19 and 20, the algorithm `Mono-Test-LTF`(f, ϵ) outputs "non-monotone" with probability at least $(9/10)(81/100) > 2/3$. This completes the proof. ◀

Proof of Theorem 26. From Fact 18, the number of queries used by `Regularize-and-Balance` is $\tilde{O}(\log^{41} n/\epsilon^{90})$, since the main bottleneck is the call to `Find-Hi-Influence-Vars`. In `Main-Procedure`, the bottleneck is the $O(\log n)$ calls to `Find-Hi-Influence-Vars` in `Maintain-Regular-and-Balance`, each of query complexity $\tilde{O}(\log^{41} n/\epsilon^{90})$, despite the slightly different parameters. Note that we run the edge tester when there are fewer than $1/\tau^2$ many stars, so it makes $\tilde{O}\left(\log^4 n/\epsilon^9\right)$ many queries. ◀

On Axis-Parallel Tests for Tensor Product Codes

Alessandro Chiesa[1], **Peter Manohar**[2], **and Igor Shinkar**[3]

1 **UC Berkeley, Berkeley, CA, USA**
alexch@berkeley.edu
2 **UC Berkeley, Berkeley, CA, USA**
manohar@berkeley.edu
3 **UC Berkeley, Berkeley, CA, USA**
igors@berkeley.edu

Abstract

Many low-degree tests examine the input function via its restrictions to random hyperplanes of a certain dimension. Examples include the line-vs-line (Arora, Sudan 2003), plane-vs-plane (Raz, Safra 1997), and cube-vs-cube (Bhangale, Dinur, Livni 2017) tests.

In this paper we study tests that only consider restrictions along *axis-parallel* hyperplanes, which have been studied by Polishchuk and Spielman (1994) and Ben-Sasson and Sudan (2006). While such tests are necessarily "weaker", they work for a more general class of codes, namely tensor product codes. Moreover, axis-parallel tests play a key role in constructing LTCs with inverse polylogarithmic rate and short PCPs (Polishchuk, Spielman 1994; Ben-Sasson, Sudan 2008; Meir 2010). We present two results on axis-parallel tests.

1. *Bivariate low-degree testing with low-agreement.* We prove an analogue of the Bivariate Low-Degree Testing Theorem of Polishchuk and Spielman in the low-agreement regime, albeit with much larger field size. Namely, for the 2-wise tensor product of the Reed–Solomon code, we prove that for sufficiently large fields, the 2-query variant of the axis-parallel line test (row-vs-column test) works for *arbitrarily small agreement*. Prior analyses of axis-parallel tests assumed high agreement, and no results for such tests in the low-agreement regime were known.

   Our proof technique deviates significantly from that of Polishchuk and Spielman, which relies on algebraic methods such as Bézout's Theorem, and instead leverages a fundamental result in extremal graph theory by Kővári, Sós, and Turán. To our knowledge, this is the first time this result is used in the context of low-degree testing.
2. *Improved robustness for tensor product codes.* Robustness is a strengthening of local testability that underlies many applications. We prove that the axis-parallel hyperplane test for the m-wise tensor product of a linear code with block length n and distance d is $\Omega(\frac{d^m}{n^m})$-robust. This improves on a theorem of Viderman (2012) by a factor of $1/\operatorname{poly}(m)$. While the improvement is not large, we believe that our proof is a notable simplification compared to prior work.

1998 ACM Subject Classification E.4 Coding and Information Theory

Keywords and phrases tensor product codes, locally testable codes, low-degree testing, extremal graph theory

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.39

1 Introduction

Locally testable codes (LTCs) are error-correcting codes for which, given an input word, one can verify whether the word belongs to or is far from the code by inspecting the word in a few random locations. LTCs have been studied extensively in different contexts,

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 39; pp. 39:1–39:22
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

including program checking, interactive proofs, and probabilistically checkable proofs (PCPs) [17, 30, 5, 4, 28, 22]. Goldreich and Sudan [22] describe LTCs as "combinatorial counterparts of the complexity theoretic notion of PCPs", motivating the study of these objects separately.

LTC constructions

The first constructions of LTCs were algebraic in nature, and relied on multivariate polynomials. Starting with the seminal work of Blum, Luby, and Rubinfeld [17], there has been much work on such algebraic LTCs by way of results on *linearity testing* and *low-degree testing* in numerous settings [17, 7, 12, 6, 1, 16]. Many other constructions [26, 33, 24] further optimize parameters of these codes, including rate, distance, and the number of queries made by the tester.

Ben-Sasson and Sudan [10] suggested a *combinatorial* approach to construct LTCs starting from any linear code by

(i) applying the *tensor product* operation [34, 35] to the code, and
(ii) testing the resulting code via the *axis-parallel hyperplane test.*

We now discuss both.

The 2-wise tensor product of a linear code $C \subseteq \mathbb{F}^n$, denoted C^2, is the code in $\mathbb{F}^{n^2}$ consisting of all 2-dimensional matrices whose n rows and n columns are codewords in C; similarly, the m-wise tensor product of C, denoted C^m, is the code in $\mathbb{F}^{n^m}$ consisting of all m-dimensional matrices M whose restrictions to any axis-parallel $(m-1)$-dimensional hyperplane is a codeword in C^{m-1}. For example, the code of evaluations of all m-variate polynomials of individual degree at most r is the m-wise tensor product of the code of evaluations of all univariate polynomials of degree at most r.

The axis-parallel hyperplane test for the code C^m works as follows: given a word M, sample a random axis-parallel hyperplane and check if the restriction of M to this hyperplane is a codeword in C^{m-1}. This natural test extends ideas of axis-parallel line tests used in early PCP constructions [5, 4, 2] to arbitrary tensor product codes.

We study two aspects of the axis-parallel hyperplane test for tensor product codes.

(1) Low-agreement regime

All of the aforementioned works study the axis-parallel hyperplane test in the "high-agreement regime", in which the given codeword is within the unique decoding radius of the tensor product code. What can be said about the "low-agreement regime", in which the given codeword may be as far as the list-decoding radius? This setting is more challenging because one wishes to deduce that a given word has some noticeable global correlation with a codeword, or a short list of codewords, by only assuming that local views of the test have some non-trivial agreement with accepting views (but may not necessarily be very close to such views).

Results in the low-agreement regime are known for *other* tests, such as tests for the Hadamard code [6] and the long code [23] as well as random *non*-axis-parallel hyperplane tests in various dimensions [29, 3, 27]. Moreover, these have applications to PCP constructions and hardness of approximation. However, to our knowledge *prior to our work no results are known for the low-agreement regime of axis-parallel tests.*

(2) Robustness

Ben-Sasson and Sudan [10] analyze the axis-parallel hyperplane test via the notion of *robustness*, a stronger notion of local testability borrowed from the PCP literature [9, 19].

Informally, a test for a code is robust if, given any input that is far from the code, the local view of the test is also far from an accepting view on average. For example, the axis-parallel hyperplane test is robust if, given any M that is far from C^m, the restriction of M to a random hyperplane is far from C^{m-1} on average.

Robustness thus relates the global distance to the expected local view distance and, as shown in [10], facilitates query reduction via a natural way to compose tests; this notion has also found applications to proof composition in the setting of PCPs [9]. These works have motivated the study of the robustness of the axis-parallel hyperplane test for tensor product codes, establishing both positive results [20, 13, 14, 32] and limitations [31, 18, 21].

Despite significant progress, robustness results for the axis-parallel hyperplane test seem to be *far from tight*. The best known relation between the global distance and the local distance is due to Viderman [32], but no examples that come anywhere close to his proven bound are known.

2 Main results

We present two main results about tests for tensor product codes. First, we prove an analogue of the Bivariate Low-Degree Testing Theorem of Polishchuk and Spielman [28] in the low-agreement regime, albeit with much larger field size. Second, we improve on the robustness of the hyperplane test for testing the tensor product code C^m, for $m \geq 3$. We now discuss our results.

2.1 Bivariate low-degree testing in the low-agreement regime

One of the applications of locally testable codes is constructing PCPs, where it is often desirable to reduce the number of queries made by the test. Typically this is done by increasing the alphabet size so that each “large” symbol bundles together several “small” symbols from different locations of the given word. This bundling now introduces a *consistency* problem, because two large symbols may in principle disagree about the same location in the word.

For example, in [29, 3, 27, 15] the test has access to (alleged) restrictions of a low-degree polynomial to all lines, planes, cubes, or other low-degree manifolds. The test samples several queries that intersect, and checks that their answers are consistent on the intersection. These works establish that if the test accepts with probability above a certain threshold, then the restrictions are close to the restrictions of some low-degree polynomial.

We study this problem in a modified setting, where the test only has access to *axis-parallel* restrictions. Restricting the test in this way makes its task more difficult, but doing so provides other advantages. First, axis-parallel restrictions are sometimes the only natural restrictions, such as when testing the m-wise tensor product of a general linear code C (one may consider restrictions to all $(m-1)$-dimensional hyperplanes). Second, having fewer restrictions enables more efficiency, e.g., it facilitates the construction of short PCPs [28, 11].

Indeed, for this very reason, Polishchuk and Spielman [28] study the above problem for bivariate polynomials, where $m = 2$ and C is the degree-r Reed–Solomon code. That is, the test has access to a table of row polynomials and a table of column polynomials, and its goal is to check if these are consistent with restrictions of a bivariate polynomial of individual degree r. The test works by as follows: pick a random $(x, y) \in \mathbb{F}^2$, read the row and column polynomials through this point, and accept if and only if the two polynomials are equal on (x, y).

Clearly, if all the row polynomials and column polynomials are restrictions of a bivariate polynomial of individual degree r, then the test always accepts. They prove that, conversely, if the test accepts with probability close to 1, then the given polynomials are "close" to being restrictions (to axis-parallel lines) of some low-degree bivariate polynomial, as written below. In the statement, we say that a bivariate polynomial in variables x and y has degree (a, b) if the degree in x is at most a and that in y is at most b. This means that the table of row polynomials, $\mathcal{R}(x, y)$, has degree (r, n) and the table of column polynomials, $\mathcal{C}(x, y)$, has degree (n, r), where n is the size of the table.

▶ **Theorem 1** ([28]). *Let $\mathbb{F}$ be a field and $X, Y \subseteq \mathbb{F}$ subsets of size $n := |X| = |Y|$. Let $\mathcal{R}(x, y)$ be a polynomial of degree (r, n) and $\mathcal{C}(x, y)$ a polynomial of degree (n, r) such that*

$$\Pr_{(x,y)\in X\times Y}[\mathcal{C}(x, y) = \mathcal{R}(x, y)] = 1 - \gamma^2$$

for some $\gamma > 0$. If $n > 2\gamma n + 2r$, then there exists a polynomial $Q(x, y)$ of degree (r, r) such that

$$\Pr_{(x,y)\in X\times Y}[\mathcal{C}(x, y) = \mathcal{R}(x, y) = Q(x, y)] \geq 1 - 2\gamma^2 \enspace .$$

The theorem above assumes that $n > 2\gamma n+2r$, which means that $\gamma^2 < (1/2-r/n)^2 < 1/4$. In other words, it requires the row polynomials and column polynomials to agree on (at least) more than three quarters of the points in $X \times Y$. A slight improvement in the parameters of this theorem is shown in [8]. However, their result still requires the polynomials to agree on a large fraction of the points in $X \times Y$. But what, if anything, can be said if we only assume that they agree, for example, on more than a 0.1-fraction of those points?

There are several results on low-degree testing that show that, even if we only assume that the test accepts with noticeable probability (for the row-vs-column test this probability equals the agreement between row and column polynomials), one can *still* prove the existence of a *short list* of polynomials that 'explain' most of this probability, and this in turn has applications to constructing PCPs with small errors (see, e.g., [29, 3, 27]).

Our next result gives a positive answer to the question above, stating that even in the low-agreement regime, we can still deduce some structure about the polynomials $\mathcal{R}$ and $\mathcal{C}$, assuming that the field size is sufficiently large.

▶ **Theorem 2.** *Let $\mathbb{F}$ be a field of size n, $r \in \mathbb{N}$, and $\delta, \varepsilon \in \mathbb{R}$ be such that $\delta > \varepsilon > 6\sqrt{r/n}$. Let $\mathcal{R}(x, y)$ be a polynomial of degree (r, n) and $\mathcal{C}(x, y)$ a polynomial of degree (n, r) such that*

$$\Pr_{(x,y)\in \mathbb{F}^2}[\mathcal{C}(x, y) = \mathcal{R}(x, y)] = \delta \enspace .$$

If $n > \exp(\Omega(\frac{r}{\varepsilon}\log(\frac{1}{\varepsilon})))$, then there exist $t = O(\frac{1}{\varepsilon})$ polynomials $Q_1(x, y), \ldots, Q_t(x, y)$ of degree (r, r) such that

$$\Pr_{(x,y)\in \mathbb{F}^2}[\exists i \in [t] \ \mathcal{C}(x, y) = \mathcal{R}(x, y) = Q_i(x, y)] \geq \delta - \varepsilon \enspace .$$

We remark that Theorem 2 holds in general for the 2-wise tensor of *any* linear code $C \subseteq \mathbb{F}^n$ with minimal distance $\geq n - r$ such that $n > \exp(\Omega(\frac{r}{\varepsilon}\log(\frac{1}{\varepsilon})))$. In particular, this means that the minimal distance of C is at least $n - O(\log n)$. See the paragraph *Beyond polynomials* below for details.

Note that in the above theorem, δ is the agreement probability, while γ^2 in Theorem 1 is the disagreement probability. Also, since $r = o(n)$, both δ and ε can be *sub-constant.* This is the first result that analyzes the row-vs-column test in the low acceptance regime that we are aware of.

The row-vs-column test and its higher-dimensional analogues underly many known PCP constructions [5, 4, 28, 11]. However, in all these constructions the low degree tests are only analyzed in the high agreement regime. We believe that analyzing the test in the low-agreement regime may imply short PCP constructions with small (sub-constant) soundness. A weakness of the result stated in Theorem 2 is the requirement that the field size must be very large, which restricts us from getting PCPs with polynomial-size proof length. Nonetheless, we consider Theorem 2 as a promising first step in this direction. More generally, our result suggests that the low-agreement regime for tensor product codes merits further study.

To prove the theorem we leverage a fundamental result in extremal graph theory by Kövári, Sós, and Turán. To our knowledge, this is the first time this result is used in the context of low-degree testing. See Section 3.1 below for a high-level description of our proof.

Beyond polynomials

While [28]'s proof relies on polynomials (a key step is Bézout's Theorem), we rely on combinatorial techniques, so that our Theorem 2 holds in general for the 2-wise tensor of *any* linear code $C \subseteq \mathbb{F}^n$ with minimal distance $\geq n - r$ such that $n > \exp(\Omega(\frac{r}{\varepsilon}\log(\frac{1}{\varepsilon})))$. In particular, this means that the minimal distance of C is at least $n - O(\log n)$. The row-vs-column test is now given two matrices $\mathcal{R}, \mathcal{C} \in \mathbb{F}^{n \times n}$ such that every row of $\mathcal{R}$ is in C, and every column of $\mathcal{C}$ is in C. If $\Pr_{(x,y)\in[n]^2}[\mathcal{R}(x,y) = \mathcal{C}(x,y)] = \delta$, then there exist $t = O(1/\varepsilon)$ codewords $Q_1, \dots, Q_t \in C^2$ such that $\Pr_{(x,y)\in[n]^2}[\exists i \in [t] \text{ s.t. } \mathcal{R}(x,y) = \mathcal{C}(x,y) = Q_i(x,y)] > \delta - \varepsilon$.

In this context it is worth mentioning that there has been a lot of work on the robustness of the axis-parallel line test for 2-wise tensor products, proving both positive results [10, 20] and negative ones [31, 18, 21]. We find it quite remarkable that this result holds for general pairwise tensor codes, albeit with very high distance, as the closely related notion of robustness does not hold for general 2-wise tensor products.

Finally, in the high-agreement regime there is a *correspondence* between the robustness of the axis-parallel line test and the soundness of the row-vs-column test (the matrix is given as a collection of lines rather than explicitly).[1] Yet this correspondence *does not hold* in the low-agreement regime. Consider a matrix M whose rows are random independent codewords: the tensor product test passes with probability at least 0.5 (when reading a row), but M is typically far from a tensor codeword.

Open problems

We raise two questions on the low-agreement regime of axis-parallel line tests.

- *Smaller field size.* Our result (Theorem 2) assumes that the field size n is exponential in the degree r. Can one prove a similar result for smaller fields, such as $n = \text{poly}(r)$?
- *Higher dimensions.* Polishchuk and Spielman [28] explain that their result (in the high-acceptance regime) also holds in higher dimensions, where now the test is given a table of low-degree polynomials for each axis-parallel line in $\mathbb{F}^m$ and works as follows: pick a random $p \in \mathbb{F}^m$, read the polynomials along the m axis-parallel lines through p, and check that all polynomials agree on p. Can one prove a high-dimensional analogue of

[1] Let $M \in \mathbb{F}^{n \times n}$ be such that the average relative distance of a row/column of M to some codeword is $1 - \varepsilon$. One can verify that by considering the closest codewords in each row and in each column, the obtained table of row/column codewords passes the row-vs-column test with probability at least $1 - 2\varepsilon$. Therefore, there exists a tensor codeword that agrees with most of the rows and most of the columns, which in turn implies its agreement with M.

Theorem 2? Namely, is it true that if this test accepts with probability $\delta > 0$, then there is a short list of low-degree polynomials that explain most of the agreements?

2.2 Improved robustness for the axis-parallel hyperplane test

We study the robustness of the axis-parallel hyperplane test for the tensor product code $C^m \subseteq \mathbb{F}^{n^m}$, for an arbitrary linear code C with minimal distance d and block length n over the field $\mathbb{F}$. Let $\mathcal{H}$ be the test that, given a word $M \in \mathbb{F}^{n^m}$, samples a random axis-parallel $(m-1)$-dimensional hyperplane H and checks if $M|_H \in C^{m-1}$. For a word $M \in \mathbb{F}^{n^m}$, we define $\delta(M)$ to be the relative distance of the word M to the code C^m and $\rho(M)$ to be $\mathbb{E}_H[\delta(M|_H, C^{m-1})]$, the expected local distance of M. The test $\mathcal{H}$ is *α-robust* if $\rho(M) \geq \alpha \cdot \delta(M)$ for every word $M \in \mathbb{F}^{n^m}$. The 'strength' of the test increases with α, so the goal is to establish the largest α for which this inequality holds.

What is known

There are two main prior works that study the robustness of the test $\mathcal{H}$ for general m. We state the results of these works, starting with one of Ben-Sasson and Sudan [10].

▶ **Theorem 3** ([10]). *Let $C \subseteq \mathbb{F}^n$ be a linear code with minimal distance d. For $m \geq 3$ and $\left(\frac{d-1}{n}\right)^m \geq 7/8$, the test $\mathcal{H}$ is α-robust for C^m with $\alpha = 2^{-16}$.*

The above theorem is limited in that the proved robustness is small and, moreover, only provides a guarantee when C has a very large distance. Viderman [32] shows that this condition on the distance is not necessary in order to show *some* robustness guarantee.

▶ **Theorem 4** ([32]). *Let $C \subseteq \mathbb{F}^n$ be a linear code with minimal distance d. For $m \geq 3$, the test $\mathcal{H}$ is α-robust C^m with $\alpha = \frac{1}{2m^2}\left(\frac{d}{n}\right)^m$.*

The above theorem, the state of the art in this setting, improves on the previous one as

1. even if $\left(\frac{d-1}{n}\right)^m \geq 7/8$, the robustness provided by Theorem 4 is larger than that provided by Theorem 3 for $m \leq 169$;
2. a robustness guarantee is provided for any choice of m, d, n (as long as $m \geq 3$).

Our result

We present a simpler proof of Theorem 4, which also achieves a $\frac{1}{m^2}$ improvement in the robustness by showing that the hyperplane test is $\Omega(\frac{d^m}{n^m})$-robust. This improved value for the robustness appears more "natural", because $\frac{d^m}{n^m}$ is the distance of the code C^m.

▶ **Theorem 5.** *Let $C \subseteq \mathbb{F}^n$ be a linear code with minimal distance d. For $m \geq 3$, the test $\mathcal{H}$ is α-robust for C^m with $\alpha = \frac{1}{12}\left(\frac{d}{n}\right)^m$.*

Tight or not?

Several works have studied the test $\mathcal{H}$ and all resulting analyses have an exponential dependence on m in the robustness. Yet, there is no evidence indicating that this dependence is necessary. Perhaps a "dream" result of constant robustness, for all codes C and $m \geq 3$, is possible. Like previous results, we too incur the same exponential dependence in the robustness. We present some observations that may suggest that this dependence is not necessary.

- Under certain conditions on M, we can prove that $\rho(M) \geq \max\{\frac{1}{m+c}, c' \frac{d^m}{n^m}\} \cdot \delta(M)$ for constants $c, c' > 0$. These two expressions are *incomparable*, as we can set the parameters m, d, n to make either expression bigger than the other. (See Claim 25.)
- The guarantees of Theorem 3, Theorem 4, and Theorem 5 all degrade as $\frac{d^m}{n^m}$ decreases. In particular, the proven value of α in all these cases tends to 0 as $\frac{d}{n}$ tends to 0. However, if C is the Reed–Solomon code (or any other code with a similar interpolation property), then we can prove that $\delta(M) \leq \rho(M) + \frac{d}{n}$ for all M. (See Claim 27.)

We, thus, think that determining the optimal robustness of $\mathcal{H}$ is an intriguing open problem:

> What is the optimal robustness of the hyperplane test $\mathcal{H}$?
> Can one prove that $\alpha = \Omega\left(\max\left(\frac{1}{m}, \frac{d^m}{n^m}\right)\right)$, or even $\alpha = \Omega(1)$, for all codes?

In [10], [32], and our result, the proof shows that when $\rho(M)$ is below some threshold (related to the code's unique decoding radius), then $\delta(M)$ is also small. However, when $\rho(M)$ is not below this threshold, the analysis says nothing about $\delta(M)$, and naively uses $\delta(M) \leq 1$ to prove robustness in this regime. We believe that progress on understanding the optimal robustness of $\mathcal{H}$ hinges on understanding what techniques (if any) can be used to bound $\delta(M)$ in terms of $\rho(M)$ for a larger range of $\rho(M)$.

Open problems

Several intriguing questions on testing tensor product codes remain open.

- *Optimal robustness of $\mathcal{H}$.* What is the optimal robustness of the hyperplane test $\mathcal{H}$? Can one prove that $\alpha = \Omega\left(\max\left(\frac{1}{m}, \frac{d^m}{n^m}\right)\right)$, or even $\alpha = \Omega(1)$, for all codes?
- *Special cases.* Can one simplify the proof and/or prove a higher robustness if one assumes that C satisfies "nice" properties? For instance, what if C is the Reed–Solomon code (so that C^m is a Reed–Muller code of bounded individual degree)?

3 Techniques

We give an overview of the proof techniques behind Theorem 2 and Theorem 5.

3.1 Theorem 2: bivariate testing in the low agreement regime

Polishchuk and Spielman [28] prove their result (Theorem 1) using the following approach. Given $\mathcal{R}$ and $\mathcal{C}$ (as in the theorem) such that $\Pr_{x,y}[\mathcal{R}(x,y) = \mathcal{C}(x,y)] > 1 - \delta$, they define an "error polynomial" E that equals 0 for all (x,y) such that $\mathcal{R}(x,y) \neq \mathcal{C}(x,y)$. Since the fraction of points where $\mathcal{R}(x,y) \neq \mathcal{C}(x,y)$ is small, E is a low-degree polynomial. However, in the low-agreement regime that we consider, the degree of E is rather large, which seems to preclude their approach. In particular, a key step based on Bézout's Theorem in their proof appears to break down.

We take a completely different approach, which relies on a combinatorial statement from extremal graph theory. Given $\mathcal{R}$ and $\mathcal{C}$ such that $\Pr_{x,y}[\mathcal{R}(x,y) = \mathcal{C}(x,y)] = \delta$, we define $A \in \{0,1\}^{n \times n}$ to be the 'agreement matrix': $A(x,y) = 1$ if and only if $\mathcal{R}(x,y) = \mathcal{C}(x,y)$. By the assumption it follows that A has at least δn^2 ones. By invoking the Kövári-Sós-and Turán Theorem (which may be thought of as an analogue of Ramsey's Theorem for bipartite graphs) it follows that there are some $S, T \subseteq [n]$ such that $|S|, |T| > \Omega(\log(n)) \gg r$ and $A|_{S \times T} \equiv 1$. Since the rows of $\mathcal{R}$ and the columns of $\mathcal{C}$ are polynomials of degree r, we deduce that there exists a unique polynomial Q of degree (r,r) such that for all $(x,y) \in S \times T$ it holds that $\mathcal{R}(x,y) = \mathcal{C}(x,y) = Q(x,y)$.

The argument above may appear to be good progress toward our goal. However, there is a total of $\approx \delta n^2$ ones in A, and the rectangle $S \times T$ is of size $O(\log(n))$, i.e., tiny compared to n. This means that the progress is actually rather small!

Nevertheless, we can now set $A|_{S\times T}$ to be zero, and repeat the same argument again, thus covering all but a small fraction of ones of A with small rectangles. However, this raises a new problem. Each rectangle $S \times T$ found in the previous step can be *very* small, and so there are potentially many different polynomials Q that explain the agreements of $\mathcal{R}$ and $\mathcal{C}$. Our next goal is therefore to "stitch" these rectangles together to show that, in fact, there is only a *small* number of distinct polynomials. We do so by "making the rectangles larger", as we now explain.

Consider a rectangle $S \times T$ from the first step, and let $t' \in \mathbb{F} \setminus T$. Note that if there are $r + 1$ points $s' \in S$ such that $A(s', t') = 1$, then the row polynomial $\mathcal{R}(\cdot, t')$ is uniquely defined by these $r + 1$ points, and hence $A(s, t') = 1$ for all $s \in S$. Therefore, we can increase T by adding t' to it. On the other hand, if there are less that $r + 1$ such points $s' \in S$, then we may disregard these points as they amount to only a small fraction of the points (since $|S| \gg r$). Thus, on a typical rectangle $S \times T$, we can go from size $O(\log(n)) \times O(\log(n))$ to size roughly $O(\log(n)) \times \Omega(n)$.

In the last step, we show that if we have many rectangles of size $O(\log(n)) \times \Omega(n)$ then it is possible to "stitch" them together using the fact that if we have two rectangles $S_1 \times T_1$ and $S_2 \times T_2$ with corresponding polynomials Q_1 and Q_2 such that $|T_1 \cap T_2| > r$, then $Q_1 \equiv Q_2$. Indeed, this follows by the fact that if two univariate polynomials of degree r agree on more than r points, then they are equal. We then use the inclusion-exclusion principle to show that for $\varepsilon > \sqrt{\frac{2r}{n}}$ we cannot have more than $\frac{2}{\varepsilon}$ subsets $T_i \subseteq [n]$ of size at least εn such that $|T_i \cap T_j| \leq r$ for all $i \neq j$.

The full proof of Theorem 2 is provided in Section 4.

3.2 Theorem 5: improved robustness for the hyperplane test

Our goal is to prove that the axis-parallel hyperplane test $\mathcal{H}$ is α-robust for $\alpha = \frac{1}{12}\left(\frac{d}{n}\right)^m$. We prove this statement via a careful combination of the approaches taken by [10] and [32]. Specifically, we analyze $\rho(M)$ and $\delta(M)$ by studying the following combinatorial object: the *inconsistency graph* G of the hyperplane test $\mathcal{H}$, which we now informally describe.

The test $\mathcal{H}$ has access to a word $M \in \mathbb{F}^{n^m}$, allegedly in C^m. For any axis-parallel hyperplane H, we denote by g_H the closest codeword to $M|_H$ in C^{m-1} (breaking ties by picking an arbitrary closest codeword). The vertex set of the graph G is the set of $(m-1)$-dimensional hyperplanes, which are the local views of the test. There is an edge between two different hyperplanes H and H' if g_H and $g_{H'}$ disagree on the intersection of the hyperplanes, $H \cap H'$. (See Definition 10 for details.) In other words, the graph has an edge between two planes if the local codewords assigned to the planes are inconsistent. The graph G that we study is similar to the inconsistency graph analyzed in [10]. The difference is that, for some threshold parameter τ, the graph used in [10] adds an edge from H to every other H' in the graph if $\delta(M|_H, g_H) > \tau$.

First, we show that if G has a large independent set I, then there is a codeword f in C^m that agrees with the local codewords g_H on *every* hyperplane H in I. For an independent set I, we define I_b to be the set of $i \in [n]$ such that the hyperplane $\{p \in [n]^m : p_b = i\}$ is in I. A key property of tensor product codes is the unique extension property, which we formally state later on as Claim 21. Using the unique extension property of tensor product codes, we show that if there are two axes b_1 and b_2 where I_{b_1} and I_{b_2} both have at least $n - d + 1$

planes, then there is a word f in C^m where $f|_H = g_H$ for every H in the independent set. Without loss of generality assume $b_1 = 1$ and $b_2 = 2$. Intuitively, we fill in the restricted hypercube in $\mathbb{F}^{I_1 \times I_2 \times n^{m-2}}$ with the values of the closest codewords to $M|_H$ for each H in the independent set. Since the independent set is large, the restricted hypercube is large enough so that we can extend the partially filled-in hypercube to a unique codeword f in C^m. The uniqueness of the extension implies that $f|_H = g_H$ for every H in I.

Next, we analyze the structure of G to show that every edge is adjacent to a vertex of degree at least $(m-2)d/2$. The key point is that two different C^{m-2} codewords must disagree on at least d^{m-2} points, and these points have a particular structure. For two distinct C^{m-2} codewords, we prove that on each of the $m-2$ remaining axes there must be at least d planes, parallel to that axis, that contain points of disagreement. If not, then using the unique extension property we show that the two codewords must be equal, which is a contradiction. For any edge (H, H'), this gives us a total of $(m-2)d$ planes that disagree with at least one of g_H and $g_{H'}$ on $H \cap H'$, which shows that $\deg(H) + \deg(H')$ is at least $(m-2)d$. Therefore, at least one of H and H' has degree at least $(m-2)d/2$. As an immediate consequence, the set of planes with degree at least $(m-2)d/2$, which we denote by L, is a *vertex cover*, and the set of planes not in L is an *independent set* I.

With some algebraic manipulation, we relate the size of this vertex cover to the expected local distance $\rho(M)$. By expressing $\rho(M)$ as a sum over pairs of intersecting planes, we show that

$$\rho(M) \geq \frac{1}{n^m m(m-1)} \sum_{(H,H'):H\cap H' \neq \emptyset} \Delta|_{H\cap H'}(g_H, g_{H'}) \ .$$

This allows us to express the robustness of the test $\mathcal{H}$ in terms of the size of the vertex cover L.

Similar to the analysis of [32], we break up the proof into two cases. If $|L|$ is somewhat large, then $\rho(M) \geq \frac{1}{12}\left(\frac{d}{n}\right)^m$, and the theorem follows immediately because $\delta(M)$ is anyways at most 1. If $|L|$ is small, then the corresponding independent set has two axes where $|I_b| \geq n-d+1$. Therefore, there is a global codeword f that is consistent with all the hyperplanes in the independent set. We use this fact to show that $\delta(M)$ must be small when $\rho(M)$ is small, which concludes the proof.

The full proof of Theorem 5 is provided in Section 5.

4 Proof of Theorem 2

The discussions below rely on notations and statements introduced in Section A. The key step in the proof of Theorem 2 is the following lemma.

▶ **Lemma 6** (Key lemma). *Suppose that* $|\mathbb{F}| > \exp(\Omega(\frac{r}{\varepsilon}\log(\frac{1}{\varepsilon})))$. *Then, for any* $\varepsilon > \sqrt{\frac{2r}{|\mathbb{F}|}}$ *there are* $t \leq \frac{2}{\varepsilon}$ *polynomials* $Q_1, \ldots, Q_t$ *each of degree* (r, r)*, and subsets* $S_1, \ldots, S_t, B_1, \ldots, B_t \subseteq \mathbb{F}$ *such that*

1. *For all* $i \in [t]$ *and* $(x, y) \in (S_i, B_i)$ *it holds that* $\mathcal{C}(x, y) = \mathcal{R}(x, y) = Q_i(x, y)$.
2. *All* S_i*'s are pairwise disjoint.*
3. $\frac{|\cup_{i\in[t]} S_i \times B_i|}{|\mathbb{F}|^2} \geq \delta - 3\varepsilon$, *where* $\delta = \Pr[\mathcal{C}(x, y) = \mathcal{R}(x, y)]$.

Before proving Lemma 6 let us see how it immediately implies Theorem 2.

Proof of Theorem 2 using Lemma 6. Let $\varepsilon > 6\sqrt{\frac{r}{n}}$, and apply Lemma 6 with $\varepsilon/3 > \sqrt{\frac{2r}{n}}$. By Lemma 6 for some $t \leq \frac{2}{\varepsilon/3} = \frac{6}{\varepsilon}$ there are disjoint subsets $S_1 \times B_1, \dots, S_t \times B_t \subseteq \mathbb{F}^2$ such that $\frac{\left|\cup_{i\in[t]} S_i \times B_i\right|}{|\mathbb{F}|^2} \geq \delta - \varepsilon$, and for all $i \in [t]$ and $(x, y) \in (S_i, B_i)$ it holds that $\mathcal{R}(x, y) = \mathcal{C}(x, y) = Q_i(x, y)$. This implies that

$$\Pr_{(x,y)\in\mathbb{F}^2}[\exists i \in [t] \text{ s.t. } \mathcal{R}(x, y) = \mathcal{C}(x, y) = Q_i(x, y)] \geq \Pr[(x, y) \in \cup_{i\in[t]} S_i \times B_i] \ ,$$

which is at least $\delta - \varepsilon$, as required. ◀

We devote the rest of this section to proving Lemma 6.

4.1 Proof of Lemma 6

Let $n = |\mathbb{F}|$, and define the binary matrix $A \in \{0, 1\}^{n\times n}$ where $A(x, y) = 1$ if $\mathcal{C}(x, y) = \mathcal{R}(x, y)$ and $A(x, y) = 0$ otherwise. Note that by the assumption of Theorem 2, we have $\frac{\sum_{x,y\in[n]} A(x,y)}{n^2} = \delta$, i.e., the matrix A is δ-dense.

4.1.1 Step 1

In the first step we apply Theorem 19 iteratively to show that there exists a collection of disjoint sets $S_1, \dots, S_u \subseteq [n]$ with $|S_i| \geq \frac{r}{\varepsilon}$ such that for most points (x, y) it holds that if $A(x, y) = 1$, then $x \in \cup S_i$, and for each $i \in [u]$ there exists $T_i \subseteq [n]$ of size $|T_i| \geq \frac{r}{\varepsilon}$ such that $A_{S_i\times T_i} \equiv 1$.

▶ **Claim 7.** *Let $n, r \in \mathbb{N}$, $\delta > \varepsilon > 0$, and let $k = \lceil r/\varepsilon \rceil$. Let $A \in \{0, 1\}^{n\times n}$ be a δ-dense matrix as above, and suppose that $n > 2k^2 \left(\frac{1}{\varepsilon}\right)^{k+1}$. Then, there exist $u \in \mathbb{N}$ and two sequences $S_i \subseteq [n], T_i \subseteq [n]$ with $i = 1, \dots, u$ satisfying the following conditions.*

1. *The S_i's are pairwise disjoint.*
2. *$|S_i| = |T_i| = k$.*
3. *$A(x, y) = 1$ for every $(x, y) \in (S_i, T_i)$ and $i \in [u]$.*
4. *$\sum_{(x,y)\in([n]\setminus(\cup S_i),[n])} A(x, y) < \varepsilon n^2$.*

Proof. We will use Theorem 19 to find a submatrix of A of size $k \times k$ whose entries are all 1s. By the choice of k and the assumption that n is sufficiently large we have that $(\varepsilon - \frac{k}{n})^k = \varepsilon^k(1 - \frac{k}{\varepsilon n})^k > \varepsilon^k(1 - \frac{k^2}{\varepsilon n}) > \varepsilon^k/2 > \frac{k-1}{\varepsilon n}$, and hence $\varepsilon > \sqrt[k]{\frac{k-1}{\varepsilon n}} + \frac{k}{n}$. Hence, since A is δ-dense, we have $\delta(A) \geq \delta \geq \varepsilon > \sqrt[k]{\frac{k-1}{n}} + \frac{k}{n}$. Therefore, by Theorem 19 there exist $S_1 \subseteq [n], T_1 \subseteq [n]$ each of size $|S_1| = |T_1| = k$ such that $A|_{S_1\times T_1} \equiv 1$.

Next, we remove the rows contained in S_1 from A, and apply the same argument again. Let $M_1 = [n] \setminus S_1$ and define A_1 to be the $(n - k) \times n$ submatrix of A whose rows are indexed by M_1. Note that if $\sum_{x\in M_1,y\in[n]} A_1(x, y) > \varepsilon n^2$ then $\delta(A_1) \geq \frac{\varepsilon n}{|M_1|}$, and thus we have $\delta(A_1) \geq \frac{\varepsilon n}{n-k} > \varepsilon > \sqrt[k]{\frac{k-1}{|M_1|}} + \frac{k}{n}$. Therefore, we can apply Theorem 19 again, and find $S_2 \subseteq M_1$ and $T_2 \subseteq [n]$ of size $|S_2| = |T_2| = k$ such that $A|_{S_2\times T_2} \equiv 1$.

We repeat the same argument again, for each $i \geq 2$ defining the the subset $M_i = M_{i-1} \setminus S_{i-1}$, and letting $A_i = A_{M_i\times[n]}$. Note that if $\sum_{x\in M_i,y\in[n]} A(x, y) \geq \varepsilon n^2$ then $|M_i| \geq \varepsilon n$, and $\delta(A_i) \geq \frac{\varepsilon n}{|M_i|} \geq \varepsilon > \sqrt[k]{\frac{k-1}{|M_i|}} + \frac{k}{n}$. Therefore, by Theorem 19 there exist $S_i \subseteq M_i$ and $T_i \subseteq [n]$ of size $|S_i| = |T_i| = k$ such that $A|_{S_i\times T_i} \equiv 1$.

We stop the process after u iterations when $\sum_{x\in M_u,y\in[n]} A(x, y) < \varepsilon n^2$. By definition of the S_i's and T_i's, this gives us the subsets with the desired properties. ◀

By the assumption $|\mathbb{F}| = n > \exp(\Omega(\frac{r}{\varepsilon}\log(\frac{1}{\varepsilon})))$ in Theorem 2 we have $n > 2k^2\left(\frac{1}{\varepsilon}\right)^{k+1}$. Therefore, we can apply Claim 7 on A to get S_i's and T_i's as in the claim.

4.1.2 Step 2

Next, we show that the sets T_i in the previous step can be chosen to be of size at least εn.

▶ **Claim 8.** *Let $\{(S_i, T_i)\}_{i=1}^u$ be the sets from Claim 7. For each $i \in [u]$ define $B_i = \{y_0 \in [n] : \sum_{x\in S_i} A(x, y_0) \geq r + 1\}$. Then*
1. $\sum_{i\in[u]}\sum_{\substack{x\in S_i\\ y\in[n]\setminus B_i}} A(x, y) \leq \varepsilon n^2$.
2. *For every $i \in [u]$ if $y_0 \in B_i$ then $A(x, y_0) = 1$ for all $x \in S_i$.*

Proof. The first item is by the choice of $k \geq r/\varepsilon$. In each $i \in [u]$ and $y \in [n] \setminus B_i$ it holds that less than ε fraction of the entries are ones, and hence the total number of ones in all $i \in [u]$ and $y \in [n] \setminus B_i$ is less that εn^2. Formally, we have

$$\sum_{i\in[u]}\sum_{\substack{x\in S_i\\ y\in[n]\setminus B_i}} A(x, y) \leq \sum_{i\in[u]}\sum_{y\in[n]\setminus B_i} r \leq u \cdot n \cdot r \leq \varepsilon n^2 \ ,$$

where the last inequality uses the fact that $u \leq n/k$, and $k \geq r/\varepsilon$.

To prove the second item, we use Corollary 17. Suppose that $A(x_0, y_0) = 0$ for some $x_0 \in S_i$ and $y_0 \in B_i$. By the assumption on B_i, it holds that $|\{x \in S_i : A(x, y_0) = 1\}| \geq r+1$. Let $S = \{x_0\} \cup \{x \in S_i : A(x, y_0) = 1\}$, and let $T = \{y_0\} \cup T_i$, so that $A(x, y) = 1$ for all $(x, y) \in S\times T \setminus \{(x_0, y_0)\}$. Recall that, by definition of A, $\mathcal{R}(x, y) = \mathcal{C}(x, y)$ for all such (x, y), and hence, by Corollary 17 we also have $\mathcal{R}(x_0, y_0) = \mathcal{C}(x_0, y_0)$, and thus $A(x_0, y_0) = 1$. ◀

Note that the ones not covered by $\cup_i(S_i \times B_i)$ are the $\leq \varepsilon n^2$ ones omitted in Claim 7 and the $\leq \varepsilon n^2$ ones disregarded in the proof of Claim 8 above. Let us also disregard all S_i's and B_i's such that $|B_i| \leq \varepsilon n$, and consider only the remaining subsets. Note that the set of B_i's with $|B_i| \leq \varepsilon n$ can contain at most εn^2 ones. Redefining u to be the number of remaining sets, we get two collections of subsets $\{S_i \subseteq [n], B_i \subseteq [n]\}_{i=1}^u$ such that
1. the S_i's are pairwise disjoint.
2. $|B_i| > \varepsilon n$ for all $i \in [u]$.
3. $\sum_{(x,y)\in\cup_{i=1}^u S_i\times B_i} \geq (\delta - 3\varepsilon)n^2$.
4. $A|_{S_i\times B_i} \equiv 1$ for all $i \in [u]$.

In particular, by Lemma 16 for each $i = 1, \ldots, u$ there is a polynomial P_i of degree (r, r) such that $\mathcal{R}(x, y) = \mathcal{C}(x, y) = P_i(x, y)$ for all $(x, y) \in S_i \times B_i$.

4.1.3 Step 3

Next, we observe that if two sets B_i, B_j from the previous step have large intersection, then the corresponding polynomials P_i and P_j are equal.

▶ **Claim 9.** *Suppose that $|B_i \cap B_j| \geq r+1$ for some $i \neq j \in [u]$. Then $P_i = P_j$ and $B_i = B_j$.*

Proof. Denote $B = B_i \cap B_j$. Note that, for each $y \in B$, $P_i(x, y) = \mathcal{C}(x, y)$ for all $|S_i| = k > r + 1$ values of $x \in S_i$, and hence $P_i(x, y) = \mathcal{C}(x, y)$ for all $x \in [n]$. In particular, $P_i(x, y) = \mathcal{C}(x, y)$ for all $(x, y) \in S_j \times B$. Therefore, $P_i|_{S_j\times B} \equiv P_j|_{S_j\times B}$, and thus $P_i \equiv P_j$ by Lemma 15. Applying Corollary 17, we conclude that $P_i(x, y) = P_j(x, y) = \mathcal{C}(x, y) = \mathcal{R}(x, y)$ for all $(x, y) \in (S_i \cup S_j) \times (B_i \cup B_j)$. This implies that $B_i = B_j$, as required. ◀

4.1.4 Completing the proof

In the last step we will show that there is a short list of $t \leq \frac{2}{\varepsilon}$ polynomials $Q_1, \ldots, Q_t$ such that each of the P_i's is in fact equal to one of the Q_j's. Indeed, denote the number of different B_i's by t. By Claim 9, if $B_i \neq B_j$ then $|B_i \cap B_j| \leq r$, and thus by the inclusion-exclusion principle we have

$$n \geq \left|\cup_{i=1}^{t} B_i\right| \geq \sum_{i=1}^{t} |B_i| - \sum_{i \neq j} |B_i \cap B_j| \geq t \cdot \varepsilon n - \binom{t}{2} r \ ,$$

where in the last inequality we used the bound $|B_i| > \varepsilon n$ for all i. If $t \geq \frac{2}{\varepsilon}$, then $n \geq t \cdot \varepsilon n - \binom{t}{2} r \ \geq 2n - \frac{2}{\varepsilon^2} r$, and thus $\varepsilon < \sqrt{\frac{2r}{n}}$, which contradicts the assumption on ε. Therefore $t < \frac{2}{\varepsilon}$, as required.

5 Proof of Theorem 5

We prove Theorem 5. The discussions below rely on notations and statements introduced in Section B.

Let C be a linear code with distance d and block length n over $\mathbb{F}$, and let C^m be the m-wise tensor product of C, for some $m \geq 3$. Let M be the input to the test $\mathcal{H}$, which is an evaluation table of a function from $[n]^m \to \mathbb{F}$. Define g_H to be the closest C^{m-1} word to $M|_H$, where ties are broken by picking an arbitrary closest codeword. We will view M as fixed throughout the analysis.

We need to show that $\rho(M) \geq \alpha \cdot \delta(M)$, for $\alpha = \frac{1}{12}\left(\frac{d}{n}\right)^m$. The main idea in the proof is to upper bound $\delta(M)$ by figuring out how to "stitch" together the g_H's to make a global codeword f. We begin by defining the inconsistency graph G. The graph G has each hyperplane as a vertex, and has an edge between two hyperplanes H and H' if they have nonzero intersection and their respective local codewords g_H and $g_{H'}$ are inconsistent, i.e., they disagree on some point p in their intersection $H \cap H'$.

▶ **Definition 10** (Inconsistency Graph)**.** The inconsistency graph G of the test $\mathcal{H}$ is a graph where V is the set of hyperplanes, and $E = \{(H, H') : \exists p \in H \cap H' \text{ s.t. } g_H(p) \neq g_{H'}(p)\}$.

The proof will be divided into several steps. First, we will show that if G contains a large independent set, namely a large set of planes which are all consistent with each other, then there is a global codeword f that stitches together all of the local codewords g_H for every H in the independent set. Then, we will show that every edge in G is adjacent to a vertex of (somewhat) large degree. This will imply that the set of vertices that have large degree is a vertex cover, and its complement is an independent set. We will then show that $\rho(M)$ is lower bounded by some function that is linear in the number of vertices that have large degree. Using these components, we will conclude the proof.

5.1 Step 1: the case of a large independent set

We will show that if G has a large independent set I, then there is an f in C^m that agrees with g_H on H for every H in I. In other words, f is the codeword of C^m that stitches together all of the g_H's in the independent set. The proof relies on Claim 21.

▶ **Lemma 11** (Interpolation)**.** *If G has an independent set I of size $|I| > (m-1)(n-d) + n$, then there exists f in C^m such that $f|_H = g_H$ for every $H \in I$.*

Our proof of this lemma is similar to the proof of a different lemma in [10].

Proof. Define I_b to be the set of $i \in n$ such that the plane (b, i) is in I. Since $|I| > (m-1)(n-d)+n$, there must exist $b_1 \neq b_2$ such that $|I_{b_1}|$ and $|I_{b_2}|$ are at least $n-d+1$, as otherwise $|I| = \sum_{b=1}^{m} |I_b| \leq (m-1)(n-d)+n$. Without loss of generality assume $b_1 = 1$ and $b_2 = 2$. Let $S = I_1 \times I_2 \times [n]^{m-2}$ and let $g \colon S \to \mathbb{F}$ be a matrix in $\mathbb{F}^S$. Define $g(p) = g_H(p)$ for every $p \in S$, where H is some plane in $I_1 \cup I_2$ such that $p \in H$. Note that g is well-defined since all the planes in I are consistent with each other, as I is an independent set.

We claim that $g \in C|_{I_1} \otimes C|_{I_2} \otimes C^{m-2}$. This is because for any $H \in I_1$ it holds that $g|_H \in C|_{I_2} \otimes C^{m-2}$, as $g|_H = g_H$ except that the second axis is now restricted to I_2. This means that for every axis $b \neq 1, 2$ and for every line ℓ_b parallel to the b-th axis it holds that $g|_{\ell_b} \in C$. Also, for every line ℓ_2 parallel to the second axis we have that $g|_{\ell_2} \in C|_{I_2}$, because we took a C^{m-1} codeword and restricted it to the subset I_2. However, by symmetry we can repeat the same argument, swapping axis 1 and axis 2, and hence for every line ℓ_1 parallel to the first axis it must hold that $g_{\ell_1} \in C|_{I_1}$. Thus, $g \in C|_{I_1} \otimes C|_{I_2} \otimes C^{m-2}$. Since $|I_1|$ and $|I_2|$ are at least $n-d+1$, we can apply Claim 21 to the code $C|_{I_1} \otimes C|_{I_2} \otimes C^{m-2}$ to extend g to a unique codeword $f \in C^m$.

We still need to show that $f|_H = g_H$ for every $H \in I$. By definition of C^m we have $f|_H \in C^{m-1}$. There are three cases. If $H \in I_1$, then f agrees with g_H on a subset of H of size $I_2 \times [n]^{m-2}$, because $g_H|_{I_2 \times [n]^{m-2}} = g|_{I_2 \times [n]^{m-2}} = f|_{I_2 \times [n]^{m-2}}$. Similarly, if $H \in I_2$, then f agrees with g_H on a subset of size $I_1 \times [n]^{m-2}$, and if $H \in I \setminus (I_1 \cup I_2)$, then f agrees with g_H on a subset of size $I_1 \times I_2 \times [n]^{m-3}$. In all 3 of the cases, since $|I_1|$ and $|I_2|$ are at least $n-d+1$, by Claim 21 there is a unique codeword $w \in C^{m-1}$ that equals $f|_H$ (or g_H) on that subset of H. But $f|_H$ is in C^{m-1}, so by the uniqueness of the extension it follows that $f|_H = g_H$. ◀

5.2 Step 2: the structure of G

We will now show that every edge (H, H') in G is adjacent to a vertex of large degree. The proof uses the structure of C^m to show that if two planes disagree on a point, they must disagree on many points, and these points have a certain structure. Using the structure of these points, we find $(m-2)d$ planes that intersect $H \cap H'$ on at least one point that g_H and $g_{H'}$ disagree, and therefore each of these new planes must be adjacent to at least one of H and H'.

▶ **Lemma 12.** *If $(H, H') \in E$, then $\deg(H) + \deg(H') \geq (m-2)d$.*

A similar lemma appears in [10], but the graph they consider is different from ours.

Proof. Without loss of generality assume that $H = (1, i)$ and $H' = (2, j)$. Fix $k \in \{3, \dots, m\}$. Let I_k be the set of l's such that the plane (k, l) is not adjacent to both H and H'. Suppose $|I_k| \geq n-d+1$. Then $g_H|_{I_k \times [n]^{m-3}} = g_{H'}|_{I_k \times [n]^{m-3}}$. Since $|I_k| \geq n-d+1$, by Claim 21 $g_H|_{I_k \times [n]^{m-3}}$ can be extended to a unique $w \in C^{m-2}$, and so $w = g_H|_{H \cap H'}$. Similarly, $g_{H'}|_{I_k \times [n]^{m-3}}$ can be extended to a unique $v \in C^{m-2}$, and so $v = g_{H'}|_{H \cap H'}$. However, since both $g_H|_{H \cap H'}$ and $g_{H'}|_{H \cap H'}$ agree on $I_k \times [n]^{m-3}$, the uniqueness of the extension implies that they are equal, contradicting the fact that (H, H') is an edge in the graph. Therefore, $|I_k| \leq n-d$ for every k. This means that for a fixed k, there are at least d planes (k, l) such that g_H and $g_{H'}$ disagree on the intersection of all 3 planes. Since g_H and $g_{H'}$ disagree, $g_{(k,l)}$ can agree with at most one of them, so at least one of $(H, (k, l))$ and $(H', (k, l))$ is an edge. This holds for at least d planes for every k, which is a total of $(m-2)d$ planes. Therefore, $\deg(H) + \deg(H') \geq (m-2)d$. ◀

Thus, for every edge (H, H') one of H and H' has degree $\geq (m-2)d/2$, so we deduce the following corollary.

▶ **Corollary 13** (Vertex Cover). *The set L of vertices with degree $\geq (m-2)d/2$ is a vertex cover.*

5.3 Step 3: relating the expected local distance to the vertex cover

We now relate the set of vertices of large degree to the expected local view distance of the test $\mathcal{H}$. The main idea is to put the expression for $\rho(M)$ into a particular form, and then apply the triangle inequality to express $\rho(M)$ as a sum over edges in the graph. Using a simple relation between $|L|$ and $|E|$, the lemma follows.

▶ **Lemma 14.** *Let L be the set of vertices with large degree. Then $\rho(M) \geq \frac{m-2}{4(m-1)} \frac{d^{m-1}}{n^{m-1}} \frac{|L|}{nm}$.*

Proof. By definition, $\rho(M) = \frac{1}{n^m m} \sum_H \Delta(M|_H, g_H)$. For any $H = (b, i)$,

$$\Delta(M|_H, g_H) = \frac{1}{m-1} \sum_{c \in [m] \setminus \{b\}} \sum_{j \in [n]} \Delta|_{H \cap (c,j)}(M, g_H) = \frac{1}{m-1} \sum_{H' : H \cap H' \neq \emptyset} \Delta|_{H \cap H'}(M, g_H) \ .$$

This is because for any point $p \in H$ and for any axis $c \neq b$, the point p is in the intersection $H \cap (c, j)$ for exactly one j. Therefore,

$$\rho(M) = \frac{1}{n^m m} \sum_H \Delta(M|_H, g_H) = \frac{1}{n^m m} \sum_H \frac{1}{m-1} \sum_{H' : H \cap H' \neq \emptyset} \Delta|_{H \cap H'}(M, g_H)$$

Every pair (H, H') with $H \cap H' \neq \emptyset$ appears exactly twice in the sum, contributing $\Delta|_{H \cap H'}(M, g_H)$ and $\Delta|_{H \cap H'}(M, g_{H'})$ to the sum. Therefore,

$$\begin{aligned}\rho(M) &= \frac{1}{n^m m(m-1)} \sum_{(H,H') : H \cap H' \neq \emptyset} \Delta|_{H \cap H'}(M, g_H) + \Delta|_{H \cap H'}(M, g_{H'}) \\ &\geq \frac{1}{n^m m(m-1)} \sum_{(H,H') : H \cap H' \neq \emptyset} \Delta|_{H \cap H'}(g_H, g_{H'}) = \frac{1}{n^m m(m-1)} \sum_{(H,H') \in E} \Delta|_{H \cap H'}(g_H, g_{H'}) \ .\end{aligned}$$

as $(H, H') \notin E \implies \Delta|_{H \cap H'}(g_H, g_{H'}) = 0$ by definition. Fix $(H, H') \in E$. The local codewords g_H and $g_{H'}$ are both in C^{m-1}, so $g_H|_{H \cap H'}$ and $g_{H'}|_{H \cap H'}$ are both C^{m-2} codewords. In particular, since $\Delta|_{H \cap H'}(g_H, g_{H'}) > 0$, they are *distinct* codewords, and so $\Delta|_{H \cap H'}(g_H, g_{H'}) \geq d^{m-2}$. Therefore,

$$\rho(M) \geq \frac{1}{n^m m(m-1)} \sum_{(H,H') \in E} \Delta|_{H \cap H'}(g_H, g_{H'}) \geq \frac{|E| \, d^{m-2}}{n^m m(m-1)} \ .$$

Since L is the set of vertices of degree $\geq (m-2)d/2$,

$$2 |E| = \sum_H \deg(H) \geq \sum_{H \in L} \deg(H) \geq |L| \frac{(m-2)d}{2} \implies |E| \geq |L| \frac{(m-2)d}{4} \ .$$

Thus,

$$\rho(M) \geq \frac{|E| \, d^{m-2}}{n^m m(m-1)} \geq \frac{(m-2) \, |L| \, d^{m-1}}{4 n^m m(m-1)} = \frac{(m-2)}{4(m-1)} \frac{d^{m-1}}{n^{m-1}} \frac{|L|}{nm} \ .$$ ◀

5.4 Putting things together

We are now ready to prove Theorem 5. The result follows from straightforward applications of the previous steps.

Proof of Theorem 5. If $|L| \geq (m-1)d$, then by Lemma 14 we have

$$\rho(M) \geq \frac{(m-2)}{4(m-1)} \frac{d^{m-1}}{n^{m-1}} \frac{|L|}{nm} \geq \frac{(m-2)}{4(m-1)} \frac{d^{m-1}}{n^{m-1}} \frac{(m-1)d}{nm} = \frac{m-2}{4m} \frac{d^m}{n^m} \geq \frac{m-2}{4m} \frac{d^m}{n^m} \delta(M) \ ,$$

where the last inequality holds because $\delta(M) \leq 1$. Therefore, assume that $|L| < (m-1)d$. For every f in C^m, using triangle inequality we have

$$\delta(M) \leq \delta(M,f) = \frac{1}{nm} \sum_H \delta|_H(M,f) \leq \frac{1}{nm} \sum_H \delta|_H(M,g_H) + \frac{1}{nm} \sum_H \delta|_H(g_H,f) \ .$$

Recalling that $\rho(M) = \frac{1}{nm} \sum_H \delta|_H(M,g_H)$ we get that

$$\delta(M) \leq \rho(M) + \frac{1}{nm} \sum_H \delta|_H(g_H,f) \ .$$

Since L is a vertex cover, the set $\overline{L} = V \setminus L$ is an independent set. Since $|L| < (m-1)d$, $\left|\overline{L}\right| > nm - (m-1)d = (m-1)(n-d) + n$. By Lemma 11, $\exists f^* \in C^m$ such that $f^*|_H = g_H$ for every $H \in \overline{L}$. Thus,

$$\delta(M) \leq \rho(M) + \frac{1}{nm} \sum_H \delta|_H(g_H,f^*) = \rho(M) + \frac{1}{nm} \sum_{H \in L} \delta|_H(g_H,f^*) \leq \rho(M) + \frac{|L|}{nm} \ .$$

By Lemma 14, $\rho(M) \geq \frac{(m-2)}{4(m-1)} \frac{d^{m-1}}{n^{m-1}} \frac{|L|}{nm}$. Therefore, $\frac{|L|}{nm} \leq \frac{4(m-1)n^{m-1}}{(m-2)d^{m-1}} \rho(M)$ and so

$$\delta(M) \leq \rho(M) + \frac{|L|}{nm} \leq \rho(M) \left(1 + \frac{4(m-1)n^{m-1}}{(m-2)d^{m-1}}\right) \implies \rho(M) \geq \frac{1}{1 + \frac{4(m-1)n^{m-1}}{(m-2)d^{m-1}}} \delta(M) \ .$$

Thus, $\forall M$, $\rho(M) \geq \alpha\delta(M)$, for $\alpha = \min\left(\frac{1}{1+\frac{4(m-1)n^{m-1}}{(m-2)d^{m-1}}}, \frac{m-2}{4m}\frac{d^m}{n^m}\right)$. Since $m \geq 3$, we have that $\frac{1}{1+\frac{4(m-1)n^{m-1}}{(m-2)d^{m-1}}} \geq \frac{1}{1+8\frac{n^{m-1}}{d^{m-1}}} \geq \frac{d^{m-1}}{9n^{m-1}}$ and $\frac{m-2}{4m}\frac{d^m}{n^m} \geq \frac{1}{12}\frac{d^m}{n^m}$. Therefore, $\alpha \geq \min(\frac{d^{m-1}}{9n^{m-1}}, \frac{1}{12}\frac{d^m}{n^m}) = \frac{1}{12}\frac{d^m}{n^m}$. ◀

Acknowledgements. The authors thank Eli Ben-Sasson, Oded Goldreich, and Madhu Sudan for helpful discussions, and the anonymous referees for their very useful comments. This work was supported in part by the Center for Long-Term Cybersecurity at UC Berkeley.

References

1 Noga Alon, Tali Kaufman, Michael Krivelevich, Simon Litsyn, and Dana Ron. Testing Reed-Muller codes. *IEEE Transactions on Information Theory*, 51(11):4032–4039, 2005.

2 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. Preliminary version in FOCS'92.

3 Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23(3):365–426, 2003. Preliminary version appeared in STOC'97.

4 László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, STOC'91, pages 21–32, 1991.

5 László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991. Preliminary version appeared in FOCS'90.

6 Mihir Bellare, Don Coppersmith, Johan Håstad, Marcos A. Kiwi, and Madhu Sudan. Linearity testing in characteristic two. *IEEE Transactions on Information Theory*, 42(6):1781–1795, 1996.

7 Michael Ben-Or, Don Coppersmith, Mike Luby, and Ronitt Rubinfeld. Non-abelian homomorphism testing, and distributions close to their self-convolutions. *Random Structures and Algorithms*, 32(1):49–70, 2008.

8 Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. In *Proceedings of the 45th ACM Symposium on the Theory of Computing*, STOC'13, pages 585–594, 2013.

9 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM Journal on Computing*, 36(4):889–974, 2006.

10 Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Structures and Algorithms*, 28(4):387–402, 2006.

11 Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM Journal on Computing*, 38(2):551–607, 2008. Preliminary version appeared in STOC'05.

12 Eli Ben-Sasson, Madhu Sudan, Salil Vadhan, and Avi Wigderson. Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, STOC'03, pages 612–621, 2003.

13 Eli Ben-Sasson and Michael Viderman. Tensor products of weakly smooth codes are robust. In *Proceedings of the 11th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, and of the 12th International Workshop on Randomization and Computation*, APPROX-RANDOM'08, pages 290–302, 2008.

14 Eli Ben-Sasson and Michael Viderman. Composition of semi-LTCs by two-wise tensor products. *Computational Complexity*, 24(3):601–643, 2015.

15 Amey Bhangale, Irit Dinur, and Inbal Livni Navon. Cube vs. cube low degree test. In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference*, ITCS'17, 2017.

16 Arnab Bhattacharyya, Swastik Kopparty, Grant Schoenebeck, Madhu Sudan, and David Zuckerman. Optimal testing of Reed-Muller codes. In *Property Testing – Current Research*, pages 269–275, 2010.

17 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 47(3):549–595, 1993.

18 Don Coppersmith and Atri Rudra. On the robust testability of product of codes, 2005. ECCC TR05-104.

19 Irit Dinur and Omer Reingold. Assignment testers: Towards a combinatorial proof of the PCP theorem. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'04, pages 155–164, 2004.

20 Irit Dinur, Madhu Sudan, and Avi Wigderson. Robust local testability of tensor products of LDPC codes. In *Proceedings of the 9th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, and of the 10th International Workshop on Randomization and Computation*, APPROX-RANDOM'06, pages 304–315, 2006.

21 Oded Goldreich and Or Meir. The tensor product of two good codes is not necessarily robustly testable. *Information Processing Letters*, 112(8-9):351–355, 2012.

22 Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost-linear length. *Journal of the ACM*, 53:558–655, July 2006. Preliminary version in STOC'02.

23 Johan Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.

24 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally-correctable and locally-testable codes with sub-polynomial query complexity. In *Proceedings of the 48th ACM Symposium on the Theory of Computing*, STOC'16, pages 202–215, 2016.

25 T. Kövári, V. T. Sós, and P. Turán. On a problem of Zarankiewicz. *Colloquium Mathematicae*, 3:50–57, 1954.

26 Or Meir. Combinatorial construction of locally testable codes. *SIAM Journal on Computing*, 39(2):491–544, 2009. Preliminary version appeared in STOC'08.

27 Dana Moshkovitz and Ran Raz. Sub-constant error low degree test of almost-linear size. *SIAM Journal on Computing*, 38(1):140–180, 2008. Preliminary version in STOC'06.

28 Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, STOC'94, pages 194–203, 1994.

29 Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, STOC'97, pages 475–484, 1997.

30 Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.

31 Paul Valiant. The tensor product of two codes is not necessarily robustly testable. In *Proceedings of the 8th International Workshop on Approximation, Randomization, and Combinatorial Optimization*, APPROX-RANDOM'05, pages 472–481, 2005.

32 Michael Viderman. A combination of testability and decodability by tensor products. In *Proceedings of the 15th International Workshop on Approximation, Randomization, and Combinatorial Optimization*, APPROX-RANDOM'12, pages 651–662, 2012.

33 Michael Viderman. Strong ltcs with inverse poly-log rate and constant soundness. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'13, pages 330–339, 2013.

34 Jack Keil Wolf. On codes derivable from the tensor product of check matrices. *IEEE Transactions on Information Theory*, 11(2):281–284, 1965.

35 Jack Keil Wolf and Bernard Elspas. Error-locating codes – a new concept in error control. *IEEE Transactions on Information Theory*, 9(2):113–117, 1963.

A Preliminaries for Theorem 2

A.1 Low-degree polynomials

We will use the following lemmas about low-degree polynomials in the proof of Theorem 2. These are standard interpolation lemmas, and direct proofs can be found in [28].

▶ **Lemma 15.** *Let $S, T \subseteq \mathbb{F}$ be two sets each of size at least $r+1$. Suppose that for two polynomials $Q_1(x,y), Q_2(x,y)$ of degree (r,r), if holds that $Q_1(x,y) = Q_1(x,y)$ for all $(x,y) \in (S,T)$. Then $Q_1 \equiv Q_2$.*

▶ **Lemma 16.** *Let $S, T \subseteq \mathbb{F}$ be two sets each of size at least $r+1$. Suppose that there is polynomial $\mathcal{R}(x,y)$ of degree (r,n), and a polynomial $\mathcal{C}(x,y)$ of degree (n,r) such that*

$\mathcal{R}(x,y) = \mathcal{C}(x,y)$ for all $(x,y) \in (S,T)$. Then, there exists a polynomial $Q(x,y)$ of degree (r,r) such that $Q(x,y) = \mathcal{C}(x,y) = \mathcal{R}(x,y)$ for all $(x,y) \in (S,T)$.

▶ **Corollary 17.** *Let $S,T \subseteq \mathbb{F}$ be two sets each of sizes $|S| \geq r+2$ and $|T| \geq r+2$, and let $(x_0,y_0) \in (S,T)$. Suppose that there is a polynomial $\mathcal{R}(x,y)$ of degree (r,n), and a polynomial $\mathcal{C}(x,y)$ of degree (n,r) such that $\mathcal{R}(x,y) = \mathcal{C}(x,y)$ for all $(x,y) \in (S,T) \setminus \{(x_0,y_0)\}$. Then $\mathcal{C}(x_0,y_0) = \mathcal{R}(x_0,y_0)$.*

A.2 The Kövári–Sós–Turán theorem

We first define the density of a binary matrix.

▶ **Definition 18.** Let $A \in \{0,1\}^{k\times\ell}$ be a binary matrix. Define the *density* of A to be $\delta(A) = \frac{\sum_{i\in[k],j\in[\ell]} A_{i,j}}{k\cdot\ell}$. We say that A is τ*-dense* if $\delta(A) \geq \tau$.

In the proof of Theorem 2 we will use a result due to Kövári, Sós, and Turán [25], which states that any sufficiently dense binary matrix contains a large submatrix where every entry is 1.

▶ **Theorem 19** (Kövári, Sós, Turán). *Let N, M, t, s be natural numbers that satisfy $N \geq s$ and $M \geq t \geq s$, and let $A \in \{0,1\}^{N\times M}$ be a binary matrix. If A is $\left(\sqrt[s]{\frac{t-1}{M}} + \frac{s}{N}\right)$-dense, then there are $S \subseteq [N]$ and $T \subseteq [M]$ of sizes $|S| = s$ and $|T| = t$ such that $A|_{S\times T} \equiv 1$.*

▶ Remark. The Kövári–Sós–Turán theorem is usually stated as saying that any sufficiently dense bipartite graph contains a large bipartite clique. It is clear, however, that the matrix formulation above is equivalent by associating a bipartite graph with its adjacency matrix, where the rows correspond to the vertices on the left, and the columns correspond to the vertices on the right.

B Preliminaries for Theorem 5

B.1 Linear codes

A linear code C over a field $\mathbb{F}$ is a linear subspace C of the vector space $\mathbb{F}^n$. Each codeword w in C is a string of length n, which is the block length of the code. The dimension of the code $\dim(C)$ is the dimension of C as a vector space in $\mathbb{F}^n$. For any two words w and v in $\mathbb{F}^n$, the Hamming distance between w and v, denoted by $\Delta(w,v)$, is the number of indices where i where $w_i \neq v_i$. Formally, $\Delta(w,v) = |\{i \in [n] : w_i \neq v_i\}|$. The relative distance between w and v is $\delta(w,v) = \Delta(w,v)/n$, which is the fraction of points where w and v disagree. For any subset S of $[n]$, we will define $\Delta|_S(w,v)$ to be $|\{i \in S : w_i \neq v_i\}|$, which is the Hamming distance between w and v on the subset S. Similarly, $\delta|_S(w,v) = \Delta|_S(w,v)/|S|$. The distance d of a code C is the minimum Hamming distance between any two distinct codewords of C, i.e. $d = d(C) = \min_{w\neq v\in C} \Delta(w,v)$. For any w in $\mathbb{F}^n$, the distance from w to C is defined as $\Delta(w,C) = \min_{v\in C} \Delta(w,v)$, and the relative distance is defined similarly. For any subset $S \subseteq [n]$, the distance from w to C on S is $\Delta|_S(w,C) = \min_{v\in C} \Delta|_S(w,v)$. We will write $\delta(w)$ instead of $\delta(w,C)$ when the code is clear from the context.

Linear codes have a unique extension property.

▶ **Claim 20** (Unique Extension). *Let I be a subset of $[n]$ of size at least $n-d+1$. Let C' be the restriction of the code C to the subset I. Then, for every codeword $w \in C'$ there exists a unique $v \in C$ such that $v|_I = w$.*

Proof. By definition, for every w in C' there must exist at least one v in C such that $v|_I = w$. Suppose there exists v_1 and v_2 such that $v_1|_I = v_2|_I = w$. Then v_1 and v_2 agree on S, so $\Delta(v_1, v_2) \leq n - |I| \leq d - 1$. Since v_1 and v_2 are codewords, $\Delta(v_1, v_2) < d$ if and only if $v_1 = v_2$. Therefore, the codeword w has a unique extension to C. ◀

B.2 Tensor product codes

For any linear code C, the 2-wise tensor product of C, denoted by $C^2 = C \otimes C$ is the linear code in $\mathbb{F}^{n^2}$, where every codeword $M \in \mathbb{F}^{n^2}$ is an $n \times n$ matrix whose each row and column is a codeword of C. The m-wise tensor of C, denoted by C^m, is defined recursively as $C^{m-1} \otimes C$. The code C^m has block length n^m and distance d^m. Furthermore, each $f \in C^m$ can be written as an $n \times n \times \cdots \times n$ (m times) matrix where the entries are values in $\mathbb{F}$, and each axis-parallel line is in C. It is easy to see that f is in C^m if and only if the restriction of f to any $(m-1)$-dimensional axis-parallel hyperplane H is in C^{m-1}. It is also worth noting that the fractional distance of the code C^m is $(d/n)^m$, so the fractional distance of the code decays exponentially in m.

Tensor product codes have a unique extension property that will be used many times in the proof of Theorem 5.

▶ **Claim 21** (Unique Extension for Tensor Product Codes). *Let $\{C_b\}_{b=1}^m$ be codes with blocklength n_b and distance d_b. Let $I_b \subseteq [n_b]$ be a set of size at least $n_b - d_b + 1$, and let C'_b be the projection of C_b to I_b. Then for every $w \in C' = C'_1 \otimes \cdots \otimes C'_m$, there exists a unique v in $C = C_1 \otimes \cdots \otimes C_m$ such that $v|_{I_1 \times \cdots \times I_m} = w$.*

Proof. By Claim 20, for all $b \in [m]$ the projection map $\pi_b \colon C_b \to C'_b$ is bijective. We can extend π_b to be a bijective map from the hybrid code $C'_1 \otimes \cdots \otimes C'_{b-1} \otimes C_b \otimes \cdots \otimes C_m$ to $C'_1 \otimes \cdots \otimes C'_b \otimes C_{b+1} \otimes \cdots \otimes C_m$. For any v in the first hybrid code, define $\pi_b(v) = v|_{I_1 \times \cdots \times I_b \times n_{b+1} \times \cdots \times n_m}$, which is the projection of v to I_b along the bth axis, and the identity map everywhere else. Clearly, π_b is still a bijection, and so the composition of maps $\pi = \pi_m \circ \pi_{m-1} \circ \cdots \circ \pi_1$ is therefore a bijection from C to C', which proves the claim. ◀

B.3 Locally testable codes and robust tests

A q-query test $\mathcal{T}$ for a code $C \subseteq \mathbb{F}^n$ is a probabilistic algorithm that, given oracle access to a word $w \in \mathbb{F}^n$, makes q (non-adaptive) queries to w and then accepts or rejects. Informally, C is locally testable if there is a test $\mathcal{T}$ that accepts (with probability 1) whenever w is in C, and rejects (say with probability at least 0.5) when w is far from C.

The expected local view distance $\rho^{\mathcal{T}}(w)$ of $\mathcal{T}$ on a word w is the average, over the local views of $\mathcal{T}$, of the distance of w to an accepting view. Instead of analyzing the local testability of C^m, we will instead consider a stronger notion of local testability called robustness, that was introduced in [10]. The test $\mathcal{T}$ is α-*robust* if $\rho^{\mathcal{T}}(w) \geq \alpha \cdot \delta(w, C)$ for every word $w \in \mathbb{F}^n$. The 'strength' of the test increases with α, so the goal is to establish the largest α for which this inequality holds.

B.4 The axis-parallel hyperplane test

▶ **Definition 22.** Let C be a linear code, and let C^m be the m-wise tensor of C. The axis-parallel hyperplane test $\mathcal{H}$ for C^m is the test that given a word $M \in \mathbb{F}^{n^m}$ samples a random axis-parallel $(m-1)$-dimensional hyperplane H and checks if $M|_H \in C^{m-1}$.

We introduce several observations about the test $\mathcal{H}$ that will be useful in the proof of Theorem 5. Since the hyperplanes sampled by $\mathcal{H}$ are axis-parallel, each hyperplane $H \subseteq [n]^m$ must be a set of the form $H = \{p \in [n]^m : p_b = i\}$, for some $b \in [m]$ and $i \in [n]$. This means that there are nm hyperplanes in total, and each hyperplane can be specified by the pair (b, i). We will use (b, i) to refer to the hyperplane $\{p \in [n]^m : p_b = i\}$.

For $M \in \mathbb{F}^{n^m}$ and an axis-parallel hyperplane H in $[n]^m$, we define g_H to be the closest C^{m-1} codeword to $M|_H$. If this codeword is not unique, then we break ties by picking an arbitrary closest codeword. Using this notation, the expected local view distance $\rho(M)$ can be expressed as

$$\rho(M) = \mathbb{E}_H[\delta|_H(M, C^{m-1})] = \mathbb{E}_H[\delta|_H(M, g_H)] \ ,$$

where the expectation is taken over all axis-parallel hyperplanes H.

▶ **Definition 23.** The test $\mathcal{H}$ is *α-robust* if $\rho(M) \geq \alpha \cdot \delta(M, C^m)$ for every word $M \in \mathbb{F}^{n^m}$, where $\delta(M, C^m)$ is the relative distance of the word M to the code C^m, and $\rho(M)$ is the expected local distance of M.

Note that robustness α for the test $\mathcal{H}$ is at most 1.

▶ **Lemma 24.** *The robustness of the axis-parallel hyperplane test $\mathcal{H}$ is $\alpha \leq 1$.*

Proof. Let f be any C^m codeword such that $\delta(M) = \delta(M, f)$. Then,

$$\delta(M) = \delta(M, f) = \frac{1}{nm} \sum_H \delta|_H(M, f) \geq \frac{1}{nm} \sum_H \delta|_H(M, g_H) = \rho(M)$$

since g_H is closer to $M|_H$ than $f|_H$, as $f|_H \in C^{m-1}$. Thus $\alpha \leq \rho(M)/\delta(M) \leq 1$. ◀

C Other Results

Here we will prove other results that are incomparable to Theorem 5.

We have already shown in Theorem 5 that $\mathcal{H}$ is robust for $\alpha \geq \frac{1}{12}\left(\frac{d}{n}\right)^m$. Most of the proof was dedicated to analyzing the test when the set of large degree vertices, L, was less than $(m-1)d$. In this same regime, we can prove an incomparable value for α. Specifically, we can show that for every M such that $|L| < (m-1)d$ it holds that $\rho(M) \geq \frac{1}{m+c} \cdot \delta(M)$, where c is a constant.

▶ **Claim 25.** *If $|L| < (m-1)d$, then $\rho(M) \geq \frac{1}{m+c} \cdot \delta(M)$, for $c = 32/9$. Combining with Theorem 5, this implies that $\rho(M) \geq \max\left(\frac{1}{m+c}, \frac{1}{12}\left(\frac{d}{n}\right)^m\right) \cdot \delta(M)$ when $|L| < (m-1)d$.*

Proof. Let I be the set of planes that are not in L. By the assumption $|L| < (m-1)d$, we have $|I| > (m-1)(n-d) + n$, and thus, by Lemma 11 there exists $f \in C^m$ such that $f|_H = g_H$ for all $H \in I$.

Let $K = \{p : \forall H \in I, p \notin H\}$ be the set of points that are not contained in any plane in I. Writing $I = \cup_{b=1}^m I_b$, where I_b is the set of planes (b, i) that are in I, it is clear that we can rewrite K as $K = \{p : p_b \notin I_b \ \forall b \in [m]\}$. Therefore,

$$|K| = \prod_{b=1}^m (n - |I_b|) \leq \left(n - \frac{1}{m}\sum_{b=1}^m |I_b|\right)^m = n^m \left(1 - \frac{1}{nm}\sum_{b=1}^m |I_b|\right)^m = n^m \left(\frac{|L|}{nm}\right)^m \ .$$

Now, we show that $\delta(M, f) \leq (m + c) \cdot \rho(M)$. We start by writing $\delta(M, f)$ as follows.

$$\delta(M, f) = \frac{1}{n^m}|\{p : M(p) \neq f(p)\}| = \frac{1}{n^m}|\{p \in K : M(p) \neq f(p)\}| + \frac{1}{n^m}|\{p \notin K : M(p) \neq f(p)\}| \ .$$

The first term is upper bounded by $\frac{|K|}{n^m}$, and so it is at most $\left(\frac{|L|}{nm}\right)^m$. In order to bound the second term, note that for all $p \notin K$ there exists a plane $H_p \in I$ such that $p \in H_p$, and thus, $f(p) = g_{H_p}(p)$. Therefore,

$$\begin{aligned}
\frac{1}{n^m}\left|\{p \notin K : M(p) \neq f(p)\}\right| &= \frac{1}{n^m}\left|\{p \notin K : M(p) \neq g_{H_p}(p)\}\right| \\
&\leq \frac{1}{n^m}\left|\{p \in [n]^m : M(p) \neq g_{H_p}(p)\}\right| \\
&\leq \frac{1}{n^m}\sum_{p\in[n]^m}\left|\{H : p \in H, M(p) \neq g_H(p)\}\right| \\
&= m \cdot \rho(M) \ .
\end{aligned}$$

This implies that

$$\delta(M, f) \leq \left(\frac{|L|}{nm}\right)^m + m \cdot \rho(M)$$

Next, using the bound $|L| < (m-1)d$ in the assumption of the claim, as well as the bound $\frac{|L|}{nm} \leq \rho(M) \cdot \frac{4(m-1)}{m-2} \cdot \frac{n^{m-1}}{d^{m-1}}$ from Lemma 14, we get that

$$\begin{aligned}
\delta(M, f) &\leq \left(\frac{(m-1)d}{nm}\right)^{m-1} \cdot \left(\rho(M) \cdot \frac{4(m-1)}{m-2} \cdot \frac{n^{m-1}}{d^{m-1}}\right) + m \cdot \rho(M) \\
&= \left(\left(1 - \frac{1}{m}\right)^m \cdot \frac{4m}{m-2} + m\right) \cdot \rho(M) \ .
\end{aligned}$$

For $m \geq 3$ we get that $\delta(M) \leq (m + 32/9)\rho(M)$, as required. ◀

▶ Remark. In fact, by a slightly modified argument (writing $\rho(M)$ as the sum over the intersections of k planes) we can prove that for $|L| < (m-1)d$ it holds that $\delta(M) \leq \rho(M)\left(k + c_k \frac{n^{m-k}}{d^{m-k}}\right)$, where c_k is a constant for a fixed $k \in [m]$. The proof of Theorem 5 used $k = 1$.

We can also show that when $|L| < (m-1)d$, we get a robustness of $\alpha = 1$ plus an additive term of d/n. Note that d is the distance of the code, so when $d = O(n)$, the additive term is not small.

▶ **Claim 26.** *If $|L| < (m-1)d$, then $\delta(M) \leq \rho(M) + d/n$.*

Proof. In the proof of Theorem 5, we showed that if $|L| < (m-1)d$, then

$$\delta(M) \leq \rho(M) + \frac{|L|}{nm} \leq \rho(M) + \frac{(m-1)d}{nm} \leq \rho(M) + \frac{d}{n} \ . \quad ◀$$

Next, we observe that if C is the Reed–Solomon code (or any code with a similar interpolation property), then the above holds without the constraint on $|L|$.

▶ **Claim 27.** *If C is the Reed–Solomon code, then $\delta(M) \leq \rho(M) + d/n$ unconditionally.*

Proof. Define $v_b = \sum_{H=(b,i)} \Delta|_H(M, g_H)$, and without loss of generality assume that $v_1 \leq v_2 \leq \cdots \leq v_m$. Observe that

$$\rho(M) = \frac{1}{n^m m}\sum_{b=1}^{m} v_b \ .$$

Let S be any subset of $(1,i)$ planes of size exactly $n-d+1$. By m-variate polynomial interpolation, there exists f in C^m such that $f|_H = g_H$ for every H in S. Therefore,

$$\begin{aligned}
&\delta(M) \le \delta(M,f) = \frac{1}{n^m} \sum_{H=(1,i)} \Delta|_H(M,f) \le \frac{1}{n^m} \sum_{H\in S} \Delta|_H(M,f) + \frac{1}{n^m}\left(n-|S|\right) n^{m-1} \\
&= \frac{1}{n^m} \sum_{H\in S} \Delta|_H(M,g_H) + \frac{d-1}{n} \le \frac{1}{n^m} v_1 + \frac{d-1}{n} = \frac{1}{n^m m}(m v_1) + \frac{d-1}{n} \\
&\le \frac{1}{n^m m} \sum_{b=1}^{m} v_b + \frac{d-1}{n} \le \rho(M) + \frac{d}{n} \ .
\end{aligned}$$

◀

Charting the Replica Symmetric Phase*

Amin Coja-Oghlan†1, **Charilaos Efthymiou**‡2, **Nor Jaafari**3, **Mihyun Kang**§4, **and Tobias Kapetanopoulos**¶5

1 **Goethe University, Mathematics Institute, Frankfurt, Germany**
acoghlan@math.uni-frankfurt.de
2 **Goethe University, Mathematics Institute, Frankfurt, Germany**
efthymiou@math.uni-frankfurt.de
3 **Goethe University, Mathematics Institute, Frankfurt, Germany**
jaafari@math.uni-frankfurt.de
4 **Technische Universität Graz, Institute of Discrete Mathematics, Graz, Austria**
kang@math.tugraz.at
5 **Goethe University, Mathematics Institute, Frankfurt, Germany**
kapetano@math.uni-frankfurt.de

Abstract

Random graph models and associated inference problems such as the stochastic block model play an eminent role in computer science, discrete mathematics and statistics. Based on non-rigorous arguments physicists predicted the existence of a generic phase transition that separates a "replica symmetric phase" where statistical inference is impossible from a phase where the detection of the "ground truth" is information-theoretically possible. In this paper we prove a contiguity result that shows that detectability is indeed impossible within the replica-symmetric phase for a broad class of models. In particular, this implies the detectability conjecture for the disassortative stochastic block model from [Decelle et al.: Phys. Rev. E 2011]. Additionally, we investigate key features of the replica symmetric phase such as the nature of point-to-set correlations ('reconstruction').

1998 ACM Subject Classification G.2 Discrete Mathematics, G.3 Probability and Statistics

Keywords and phrases Random factor graph, bounds for condensation phase transition, Potts antiferromagnet, diluted k-spin model, stochastic block model.

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.40

1 Introduction

1.1 The cavity method

Models based on random graphs have come to play a role in combinatorics, probability, statistics and computer science that can hardly be overstated. For example, the random k-SAT model is of fundamental interest in computer science [4], the stochastic block model has gained prominence in statistics [1, 24, 36], low-density parity check codes have become a

[*] A full version of the paper is available at https://arxiv.org/abs/1704.01043.

[†] 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.

[‡] Supported by DFG grant EF 103/1-1.

[§] Supported by Austrian Science Fund (FWF): P26826.

[¶] Supported by Stiftung Polytechnische Gesellschaft PhD grant.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 40; pp. 40:1–40:17
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

pillar of modern coding theory [40] and problems such as random graph coloring have been the lodestars of probabilistic combinatorics since the days of Erdős and Rényi [4, 10, 39]. Additionally, very similar models have been studied in statistical physics as models of disordered systems [31] and over the past 20 years physicists developed an analytic but non-rigorous technique for the study of such models called the 'cavity method'. This non-rigorous approach has inspired numerous "predictions" with an impact on an astounding variety of problems (e.g., [15, 31, 33, 42]). Hence the task of putting the cavity method on a rigorous foundation has gained substantial importance. Despite recent successes (e.g., [13, 17, 22, 36, 8, 16, 28]) much remains to be done. In particular, while the cavity method can be applied almost mechanically to a wide variety of problems, most rigorous arguments still hinge on model-specific deliberations, a state of affairs that begs the questions of whether we can rigorise the physics calculations wholesale. This is the thrust of the present paper.

One of the most important predictions of the cavity method is that random graph models generically undergo a *condensation phase transition* [27] that separates a "replica symmetric phase" without extensive long-range correlations from a phase where long-range correlations prevail. The fact *that* a phase transition occurs at the location predicted by the cavity method was recently proved for a fairly broad family of models [13]. However, that result fell short of establishing the connection to the nature of correlations claimed by the physics work. We rigorise the entire "physics story" of how correlations evolve up to the condensation phase transition as predicted in [18, 27, 29], including the nature of long-range correlations and the onset of point-to-set correlations known as the "reconstruction threshold". Furthermore, verifying a prominent prediction from [15], we prove a contiguity statement that has an impact on statistical inference problems such as the stochastic block model.

The results of this paper cover a wide class of random graph models, even broader than the family of models for which the condensation threshold was previously derived in [13]. Before presenting the general results in Section 2, we illustrate their impact on three important examples: the Potts antiferromagnet on the Erdős-Rényi random graph, the stochastic block model and the diluted k-spin model.

1.2 The Potts antiferromagnet

Let $q \geq 2$ be an integer, let $\Omega = \{1, \ldots, q\}$ be a set of q "colors" and let $\beta > 0$. The *antiferromagnetic q-spin Potts model on a graph* $G = (V, E)$ *at inverse temperature* β *is the distribution on* Ω^V defined by

$$\mu_{G,q,\beta}(\sigma) = (Z_{q,\beta}(G))^{-1} \prod_{\{v,w\}\in E} \exp(-\beta \mathbf{1}\{\sigma(v) = \sigma(w)\}), \tag{1.1}$$

where $Z_{q,\beta}(G) = \sum_{\tau\in\Omega^V} \prod_{\{v,w\}\in E} \exp(-\beta \mathbf{1}\{\tau(v) = \tau(w)\})$.

The Potts model can be viewed as a version of the graph coloring problem where monochromatic edges are not strictly forbidden but merely incur a 'penality factor' of $\exp(-\beta)$. The model has received attention in the context of the complexity of counting (e.g., [20]).

The Potts model on the random graph $\mathbb{G} = \mathbb{G}(n, p)$ with vertex set $V_n = \{x_1, \ldots, x_n\}$ whose edge set $E(\mathbb{G})$ is obtained by including each of the possible edge with probability $p \in [0, 1]$ independently, has received considerable attention as well (e.g. [5, 12, 14]). The most challenging case turns out to be that $p = d/n$ for a fixed real $d > 0$. The key problem associated with the model is to determine the distribution of the variable $\ln Z_\beta(\mathbb{G}, q, \beta)$.

Recently Coja-Oghlan, Krzakala, Perkins and Zdeborová [13] determined the *condensation threshold* $d_{\mathrm{cond}}(q, \beta)$. Specifically, this is defined as the smallest value of d where the function

$d \mapsto \lim_{n\to\infty} \frac{1}{n}\mathbb{E}[\ln Z_\beta(\mathbb{G}, q, \beta)]$ is non-analytic (the existence of the limit was proved by Bayati, Gamarnik and Tetali [9]). The precise formula for $d_{\mathrm{cond}}(q, \beta)$ is complicated and not important here, but we recall the explicit *Kesten-Stigum bound*

$$d_{\mathrm{cond}}(q,\beta) \le d_{\mathrm{KS}}(q,\beta) = \left(\frac{q-1+\mathrm{e}^{-\beta}}{1-\mathrm{e}^{-\beta}}\right)^2. \tag{1.2}$$

Moreover, Azuma's inequality shows that $\frac{1}{n}\ln Z_{q,\beta}(\mathbb{G})$ converges to $\lim_{n\to\infty}\frac{1}{n}\mathbb{E}[\ln Z_{q,\beta}(\mathbb{G})]$ in probability, and thus $\ln Z_{q,\beta}(\mathbb{G})$ has fluctuations of order $o(n)$. On the other hand, given that, e.g., the size of the largest component of $\mathbb{G}$ exhibits fluctuations of order $\sqrt{n}$ even once we condition on the number $|E(\mathbb{G})|$ of edges, one might expect that so does $\ln Z_{q,\beta}(\mathbb{G})$. Yet remarkably, the following theorem shows that $\ln Z_{q,\beta}(\mathbb{G})$ merely has *bounded* fluctuations given $|E(\mathbb{G})|$. In fact, we can determine the precise limiting distribution.

▶ **Theorem 1.** *Let $q \ge 2$, $\beta > 0$ and $0 < d < d_{\mathrm{cond}}(q,\beta)$. With $(K_l)_{l\ge3}$ a sequence of independent Poisson variables with mean $\mathbb{E}[K_l] = d^l/(2l)$, let*

$$\mathcal{K} = \sum_{l=3}^{\infty} K_l \ln(1+\delta_l) - \frac{d^l\delta_l}{2l} \qquad \text{where} \quad \delta_l = (q-1)\left(\frac{\mathrm{e}^{-\beta}-1}{q-1+\mathrm{e}^{-\beta}}\right)^l.$$

Then $\mathbb{E}|\mathcal{K}| < \infty$ and as $n \to \infty$ the random variable,

$$\ln Z_{q,\beta}(\mathbb{G}) - \left(n+\frac{1}{2}\right)\ln q - |E(\mathbb{G})|\ln\left(1-\frac{1-\mathrm{e}^{-\beta}}{q}\right)$$
$$+\frac{q-1}{2}\ln\left(1+\frac{d(1-\mathrm{e}^{-\beta})}{q-1+\mathrm{e}^{-\beta}}\right)+\frac{d\delta_1}{2}+\frac{d^2\delta_2}{4}$$

converges in distribution to $\mathcal{K}$.

Arguably the key element of the physics narrative is that for $d < d_{\mathrm{cond}}(q,\beta)$ the measure $\mu_{\mathbb{G},q,\beta}$ is free from extensive long-range correlations, while such correlations emerge for $d > d_{\mathrm{cond}}(q,\beta)$. Our next result verifies this conjecture. Formally, we define the *overlap* of two colorings $\sigma, \tau : V_n \to \Omega$ as the probability distribution $\rho_{\sigma,\tau} = (\rho_{\sigma,\tau}(s,t))_{s,t\in\Omega}$ on $\Omega\times\Omega$ with $\rho_{\sigma,\tau}(s,t) = |\sigma^{-1}(s)\cap\tau^{-1}(t)|/n$ for $s,t\in\Omega$. Thus, $\rho_{\sigma,\tau}(s,t)$ is the probability that a random vertex v is colored s under σ and t under τ. Let $\bar\rho$ denote the uniform distribution on $\Omega\times\Omega$. We write $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2$ for two independent samples from $\mu_{\mathbb{G},q,\beta}$, denote the expectation with respect to $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2$ by $\langle\,\cdot\,\rangle_{\mathbb{G},q,\beta}$ and the expectation over the choice of $\mathbb{G}$ by $\mathbb{E}[\,\cdot\,]$.

▶ **Theorem 2.** *For all $q \ge 2, \beta > 0$ we have*

$$d_{\mathrm{cond}}(q,\beta) = \inf\left\{d > 0 : \limsup_{n\to\infty} \mathbb{E}\left\langle\|\rho_{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \bar\rho\|_{\mathrm{TV}}\right\rangle_{\mathbb{G}} > 0\right\}.$$

Theorem 2 implies the absence of extensive long-range correlations in the replica symmetric phase. Indeed, for two vertices $x, y \in V_n$ and $s, t \in \Omega$ let

$$\mu_{\mathbb{G},x,y}(s,t) = \langle \mathbf{1}\{\boldsymbol{\sigma}_1(x) = s, \boldsymbol{\sigma}_1(y) = t\}\rangle_{\mathbb{G}}$$

be the joint distribution of the spins assigned to x, y. It is known (e.g., [6, Section 2]) that

$$\lim_{n\to\infty}\mathbb{E}\left\langle\|\rho_{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2}-\bar\rho\|_{\mathrm{TV}}\right\rangle_{\mathbb{G}} = 0 \qquad \text{iff} \qquad \lim_{n\to\infty}\frac{1}{n^2}\sum_{x,y\in V_n}\mathbb{E}\|\mu_{\mathbb{G},x,y}-\bar\rho\|_{\mathrm{TV}} = 0. \tag{1.3}$$

Hence, Theorem 2 implies that for $d < d_{\mathrm{cond}}(q, \beta)$, with probability tending to 1, the colors assigned to two random vertices x, y of $\mathbb{G}$ are asymptotically independent. By contrast, Theorem 2 and (1.3) also show that the same ceases to be true beyond $d_{\mathrm{cond}}(q, \beta)$.

The condensation transition is conjectured to be preceded by another threshold where certain "point-to-set correlations" emerge [27]. Intuitively, the *reconstruction threshold* is the point from where for a random vertex $\boldsymbol{y} \in V_n$ correlations between the color assigned to $\boldsymbol{y}$ and the colors assigned to *all* vertices at a large enough distance ℓ from $\boldsymbol{y}$ persist. Formally, with $\boldsymbol{\sigma}$ chosen from $\mu_{\mathbb{G}}$ let $\nabla_{\ell,}(\mathbb{G}, y)$ be the σ-algebra on Ω^{V_n} generated by the random variables $\boldsymbol{\sigma}(z)$ with z ranging over all vertices at distance at least ℓ from $\boldsymbol{y}$. Then

$$\mathrm{corr}(d) = \lim_{\ell \to \infty} \limsup_{n \to \infty} \frac{1}{n} \sum_{y \in V_n} \sum_{s \in \Omega} \mathbb{E} \left\langle \left| \left\langle \mathbf{1}\{\boldsymbol{\sigma}(y) = s\} \middle| \nabla_\ell(\mathbb{G}, y) \right\rangle_{\mathbb{G},} - 1/q \right| \right\rangle_{\mathbb{G}} \tag{1.4}$$

measures the extent of correlations between $\boldsymbol{y}$ and a random boundary condition in the limit $\ell, n \to \infty$ (the outer limit exists due to mononicity). Indeed, with the expectation $\mathbb{E}[\,\cdot\,]$ in (1.4) referring to the choice of $\mathbb{G}$, the outer $\langle\,\cdot\,\rangle_{\mathbb{G}}$ chooses a random coloring of the vertices at distance at least ℓ from y and the inner $\langle\,\cdot\,|\nabla_\ell(\mathbb{G}, y)\rangle_{\mathbb{G}}$ averages over the color of y given the boundary condition.

The *reconstruction threshold* is defined as $d_{\mathrm{rec}}(q, \beta) = \inf\{d > 0 : \mathrm{corr}_{q,\beta}(d) > 0\}$. A priori, calculating $d_{\mathrm{rec}}(q, \beta)$ appears to be quite challenging because we seem to have to control the joint distribution of all the colors at distance ℓ from y. However, according to physics predictions $d_{\mathrm{rec}}(q, \beta)$ is identical to the corresponding threshold on a random tree [27], conceptually a *much* simpler object. Formally, let $\mathbb{T}(d)$ be the Galton-Watson tree with offspring distribution $\mathrm{Po}(d)$. Let r be its root and for an integer $\ell \geq 1$ let $\mathbb{T}^\ell(d)$ be the finite tree obtained by deleting all vertices at distance greater than ℓ from r. Then

$$\mathrm{corr}^\star(d) = \lim_{\ell \to \infty} \sum_{s \in \Omega} \mathbb{E} \left\langle \left| \left\langle \mathbf{1}\{\boldsymbol{\sigma}(r) = s\} \middle| \nabla_\ell(\mathbb{T}^\ell(d), r) \right\rangle_{\mathbb{T}^\ell(d)} - 1/q \right| \right\rangle_{\mathbb{T}^\ell(d)}$$

measures the extent of correlations between the color of the root and the colors at the boundary of the tree. Accordingly, the *tree reconstruction threshold* is defined as $d^\star_{\mathrm{rec}}(q, \beta) = \inf\{d > 0 : \mathrm{corr}^\star(d) > 0\}$. Combining Theorem 2 with a result in [21], we obtain

▶ **Corollary 3.** *For every $q \geq 2$ and $\beta > 0$ we have $1 \leq d_{\mathrm{rec}}(q, \beta) = d^\star_{\mathrm{rec}}(q, \beta) \leq d_{\mathrm{cond}}(q, \beta)$.*

1.3 The stochastic block model

The disassortative *stochastic block model*, first introduced in [24], is defined as follows: First choose a random q-coloring $\boldsymbol{\sigma}^* : V_n \to \Omega$ of n vertices with $q \geq 2$. Then, setting

$$d_{\mathrm{in}} = \frac{dq\mathrm{e}^{-\beta}}{q - 1 + \mathrm{e}^{-\beta}} \quad \text{and} \quad d_{\mathrm{out}} = \frac{dq}{q - 1 + \mathrm{e}^{-\beta}} \tag{1.5}$$

we generate a random graph $\mathbb{G}^*$ by connecting any two vertices v, w of the same color with probability d_{in}/n and any two with distinct with probability d_{out}/n independently. Thus, the average degree of $\mathbb{G}^*$ converges to d in probability.

Two fundamental statistical problems arise [15]. First, given q, β, for what values of d is it possible to perform non-trivial inference, i.e., obtain a better approximation to $\boldsymbol{\sigma}^*$ given the random graph $\mathbb{G}^*$ that just a random guess (see [15] for a formal definition)? A second, more modest task is the *detection problem*, which merely asks whether the random graph $\mathbb{G}^*$ can be told apart from the natural "null model", i.e., the plain Erdős-Rényi graph $\mathbb{G}$.

Decelle, Krzakala, Moore and Zdeborová [15] predicted that for $d < d_{\mathrm{cond}}(q,\beta)$, i.e., below the Potts condensation threshold, it is information-theoretically impossible to solve either problem. On the other hand, they predicted that there exist *efficient* algorithms to solve either problem if $d > d_{\mathrm{KS}}(q,\beta)$ from (1.2). Both of these conjectures were proved in the case $q = 2$ by Mossel, Neeman and Sly [37, 38] and Massoulié [30]. The positive algorithmic conjecture was proved in full by Abbe and Sandon [2]. On the negative side, [13] shows that no algorithm can infer a non-trivial approximation to $\boldsymbol{\sigma}^*$ if $d < d_{\mathrm{cond}}(q,\beta)$ for any $q \geq 3$, $\beta > 0$. Further, Banks, Moore, Neeman, and Netrapalli [5] employed a second moment argument to determine an explicit range of d where it is impossible to discern $\mathbb{G}^*$ from $\mathbb{G}$. However, there remained an extensive gap between their explicit bound and the actual condensation threshold. Our next result closes this gap and thus settles the conjecture from [15].

$\mathbb{G}$ and $\mathbb{G}^*$ are *mutually contiguous* for $d > 0$ if for any sequence $(\mathcal{A}_n)_n$ of events we have

$$\lim_{n\to\infty} \mathbb{P}\left[\mathbb{G} \in \mathcal{A}_n\right] = 0 \quad \text{iff} \quad \lim_{n\to\infty} \mathbb{P}\left[\mathbb{G}^* \in \mathcal{A}_n\right] = 0.$$

If so, then clearly no algorithm (efficient or not) can discern with probability $1-o(1)$ whether a given graph stems from the stochastic block model $\mathbb{G}^*$ or the "null model" $\mathbb{G}$.

▶ **Theorem 4.** *For all $q \geq 3$, $\beta > 0$, $d < d_{\mathrm{cond}}(q,\beta)$ the models $\mathbb{G}$ and $\mathbb{G}^*$ are mutually contiguous.*

This result is tight since [13, Theorem 2.6] implies that $\mathbb{G}, \mathbb{G}^*$ fail to be contiguous for $d > d_{\mathrm{cond}}(q,\beta)$.

▶ Remark. There is a similar conjecture regarding the assortative version of the stochastic block model, which can be seen as an inference version of the ferromagnetic Potts model. However, the assortative block model, and ferromagnetic models generally, are beyond the scope of the present work as such models violate one of the key technical assumptions that our proofs require (condition **POS** and **BAL** below).

1.4 The diluted k-spin model

Our third application deals with a model that is of fundamental interest in physics [23, 31, 34]. For integers $k \geq 2$, $n \geq 1$ and a real $p \in [0,1]$ let $\mathbb{H} = \mathbb{H}_k(n,p)$ be the random k-uniform hypergraph on $V_n = \{x_1,\ldots,x_n\}$ whose edge set $E(\mathbb{H})$ is obtained by including each of the $\binom{n}{k}$ possible k-subsets of V_n with probability p independently. Additionally, let $\boldsymbol{J} = (\boldsymbol{J}_e)_{e\in E(\mathbb{H})}$ be a family of independent standard Gaussians. The *k-spin model* on $\mathbb{H}$ at inverse temperature $\beta > 0$ is the distribution on the set $\{-1,1\}^{V_n}$ defined by

$$\mu_{\mathbb{H},\boldsymbol{J},\beta}(\sigma) = \frac{1}{Z_\beta(\mathbb{H},\boldsymbol{J})} \prod_{e\in E(\mathbb{H})} \exp\left(\beta \boldsymbol{J}_e \prod_{y\in e} \sigma(y)\right), \tag{1.6}$$

where $Z_\beta(\mathbb{H},\boldsymbol{J}) = \sum_{\tau\in\{\pm1\}^{V_n}} \prod_{e\in E(\mathbb{H})} \exp\left(\beta \boldsymbol{J}_e \prod_{y\in e} \tau(y)\right)$.

The most interesting and at the same time most challenging scenario arises in the case of a sparse random hypergraph [32]. Specifically, set $p = d/\binom{n-1}{k-1}$ for a fixed $d > 0$.

Guerra and Toninelli [23] determined the condensation threshold in the special case where $k = 2$ but noticed that their argument does not extend to $k \geq 3$. Proving a conjecture from [19], the following theorem pinpoints the condensation thereshold for all $k \geq 3$.

Let us write $\mathcal{P}(\mathcal{X})$ for the set of all probability distributions on a finite set $\mathcal{X}$ and identify $\mathcal{P}(\mathcal{X})$ with the standard simplex in $\mathbb{R}^{\mathcal{X}}$. Moreover, let $\mathcal{P}^2(\mathcal{X})$ be the space of all probability measures on $\mathcal{P}(\mathcal{X})$ and let $\mathcal{P}^2_*(\mathcal{X})$ be the space of all $\pi \in \mathcal{P}^2(\mathcal{X})$ whose barycenter $\int_{\mathcal{P}(\mathcal{X})} \mu \mathrm{d}\pi(\mu)$ is the uniform distribution on $\mathcal{X}$. Finally, let $\Lambda(x) = x \ln x$.

▶ **Theorem 5.** *Suppose that $d > 0, \beta > 0$ and that $k \geq 3$. Let $\boldsymbol{\gamma}$ be a Poisson variable with mean d, let $\boldsymbol{I}_1, \boldsymbol{I}_2, \ldots$ be standard Gaussians and for $\pi \in \mathcal{P}^2_*(\{\pm 1\})$ let $\boldsymbol{\rho}^\pi_1, \boldsymbol{\rho}^\pi_2, \ldots \in \mathcal{P}(\{\pm 1\})$ be random variables with distribution π, all mutually independent. Define*

$$\begin{aligned}
&\mathcal{B}_{k-\mathrm{spin}}(d, \beta, \pi) \\
&\quad = \frac{1}{2}\mathbb{E}\left[\Lambda\left(\sum_{\sigma_k \in \{\pm 1\}} \prod_{j=1}^{\boldsymbol{\gamma}} \sum_{\sigma_1, \ldots, \sigma_{k-1} \in \{\pm 1\}} (1 + \tanh(\beta \boldsymbol{I}_j \sigma_1 \cdots \sigma_k)) \prod_{h=1}^{k-1} \boldsymbol{\rho}^\pi_{kj+h}(\sigma_h)\right)\right] \\
&\qquad - \frac{d}{k}\mathbb{E}\left[\Lambda\left(1 + \sum_{\sigma_1, \ldots, \sigma_k \{\pm 1\}} \tanh(\beta \boldsymbol{I}_1 \sigma_1 \cdots \sigma_k) \prod_{h=1}^{k} \boldsymbol{\rho}^\pi_h(\sigma_h)\right)\right].
\end{aligned}$$

and $d_{\mathrm{cond}}(k, \beta) = \inf\{d > 0 : \sup_{\pi \in \mathcal{P}^2_*(\{1,-1\})} \mathcal{B}_{k-\mathrm{spin}}(d, \beta, \pi) > \ln 2\}$. *Then* $0 < d_{\mathrm{cond}}(k, \beta) < \infty$ *and*

$$\lim_{n \to \infty} \frac{1}{n}\mathbb{E}[\ln Z_\beta(\mathbb{H}, \boldsymbol{J})] \begin{cases} = \ln 2 + \frac{d}{\sqrt{2\pi k}} \int_{-\infty}^{\infty} \ln(\cosh(z)) \exp(-z^2/2)\mathrm{d}z & \text{if } d \leq d_{\mathrm{cond}}(k, \beta), \\ < \ln 2 + \frac{d}{\sqrt{2\pi k}} \int_{-\infty}^{\infty} \ln(\cosh(z)) \exp(-z^2/2)\mathrm{d}z & \text{if } d > d_{\mathrm{cond}}(k, \beta). \end{cases}$$

As in the Potts model, the condensation threshold is conjectured to be related to the nature of correlations under $\mu_{\mathbb{H}, \boldsymbol{J}, \beta}$. The following theorem proves this conjecture for even values of k. We recall the overlap notation from Section 1.2.

▶ **Theorem 6.** *For all $\beta > 0$ and $k \geq 4$ even, it holds that*

$$d_{\mathrm{cond}}(k, \beta) = \inf\left\{d > 0 : \limsup_{n \to \infty} \mathbb{E}\left\langle \|\varrho_{\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2} - \bar{\rho}\|_{\mathrm{TV}} \right\rangle_{\mathbb{H}, \beta} > 0\right\}.$$

The corresponding statement for $k = 2$ was proved by Guerra and Toninelli, but they point out that their argument does not extend to larger k [23]. Furthermore, arguing as for the Potts model, we get that $\mathbb{E}\langle \|\rho_{\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2} - \bar{\rho}\|_{\mathrm{TV}} \rangle_{\mathbb{H}} = o(1)$ iff the spins of two randomly chosen vertices of $\mathbb{H}$ are asymptotically independent with probability tending to one.

2 Main results

2.1 Definitions and assumptions

Factor graphs have emerged as a unifying framework for a multitude of problems. The main results of this paper, which we present in this section, therefore deal with a general class of random factor graph models, subject merely to a few easy-to-check assumptions. Formally, let Ω be a finite set of *spins*, let $k \geq 2$ be an integer and let Ψ be a set of functions $\psi : \Omega^k \to (0, 2)$ that we call *weight functions*. A Ψ*-factor graph* $G = (V, F, (\partial a)_{a \in F}, (\psi_a)_{a \in F})$ consists of a set V of *variable nodes*, a set F of *constraint nodes*, an ordered k-tuple $\partial a = (\partial_1 a, \ldots, \partial_k a) \in V^k$ for each $a \in F$ and a weight function $\psi_a \in \Psi$ for each $a \in F$. We can picture G as a bipartite graph with variable nodes on one side and constraint nodes on the other in which each constraint node a is adjacent to $\partial_1 a, \ldots, \partial_k a$ and adorned with a weight function ψ_a. This allows us to speak of, e.g., the distance of two nodes. But we keep in mind that actually the neighborhood ∂a is an *ordered* tuple. The *Gibbs distribution* of G is the distribution on Ω^V defined by $\mu_G(\sigma) = \psi_G(\sigma)/Z(G)$ for $\sigma \in \Omega^V$, where

$$\psi_G(\sigma) = \prod_{a \in F} \psi_a(\sigma(\partial_1 a), \ldots, \sigma(\partial_k a)) \quad \text{and} \quad Z(G) = \textstyle\sum_{\tau \in \Omega^V} \psi_G(\tau).$$

For a weight function $\psi:\Omega^k\to(0,2)$ and a permutation $\theta:[k]\to[k]$ we define $\psi^\theta:\Omega^k\to(0,2)$, $(\sigma_1,\ldots,\sigma_k)\mapsto\psi(\sigma_{\theta(1)},\ldots,\sigma_{\theta(k)})$. Throughout the paper we assume that Ψ is a measurable set of weight functions such that for all $\psi\in\Psi$ and all permutations θ we have $\psi^\theta\in\Psi$. Moreover, we fix a probability distribution P on Ψ. We always denote by $\boldsymbol{\psi}$ an element of Ψ chosen from P, and we set

$$q=|\Omega|\quad\text{and}\quad\xi=\xi(P)=q^{-k}\textstyle\sum_{\sigma\in\Omega^k}\mathbb{E}[\boldsymbol{\psi}(\sigma)].$$

Furthermore, we always assume that P is such that the following three inequalities hold:

$$\begin{aligned}\mathbb{E}[\ln^8(1-\max\{|1-\boldsymbol{\psi}(\tau)|:\tau\in\Omega^k\})]&<\infty,\\ \mathbb{E}[\max\{\boldsymbol{\psi}(\tau)^{-4}:\tau\in\Omega^k\}]&<\infty,\\ \sum_{\tau\in\Omega^k}\mathbb{E}[(\boldsymbol{\psi}(\tau)-\xi)^2]&>0.\end{aligned}\tag{2.1}$$

The first two bound the 'tails' of $\boldsymbol{\psi}(\tau)$ for $\tau\in\Omega^k$. The third one provides that $\boldsymbol{\psi}$ is non-constant.

We define the random Ψ-factor graph $\boldsymbol{G}(n,m,P)$ as follows. The set of variable nodes is $V_n=\{x_1,\ldots,x_n\}$, the set of constraint nodes is $F_m=\{a_1,\ldots,a_m\}$ and the neighborhoods $\partial a_i\in V_n^k$ are chosen uniformly and independently for $i=1,\ldots,m$. Furthermore, the weight functions $\psi_{a_i}\in\Psi$ are chosen from the distribution P mutually independently and independently of $(\partial a_i)_{i=1,\ldots,m}$. Where P is apparent we just write $\boldsymbol{G}(n,m)$ rather than $\boldsymbol{G}(n,m,P)$. For a fixed $d>0$, i.e. independent of n, let $\boldsymbol{m}=\boldsymbol{m}_d(n)$ have distribution $\mathrm{Po}(dn/k)$ and write $\boldsymbol{G}=\boldsymbol{G}(n,\boldsymbol{m},P)$ for brevity. Then the expected degree of a variable node is equal to d.

Apart from the condition (2.1) the main results require (some of) the following four assumptions; crucially, they *only* refer to the distribution P on the set Ψ of weight functions.

SYM. For all $i\in\{1,\ldots,k\}$, $\omega\in\Omega$ and $\psi\in\Psi$ we have

$$\textstyle\sum_{\tau\in\Omega^k}\mathbf{1}\{\tau_i=\omega\}\psi(\tau)=q^{k-1}\xi\tag{2.2}$$

and for every permutation θ and every measurable $\mathcal{A}\subset\Psi$ we have that $P(\mathcal{A})=P(\{\psi^\theta:\psi\in\mathcal{A}\})$.

BAL. The function

$$\phi:\mu\in\mathcal{P}(\Omega)\mapsto\textstyle\sum_{\tau\in\Omega^k}\mathbb{E}[\boldsymbol{\psi}(\tau)]\prod_{i=1}^k\mu(\tau_i)$$

is concave and attains its maximum at the uniform distribution on Ω.

MIN. Let $\mathcal{R}(\Omega)$ be the set of all probability distribution $\rho=(\rho(s,t))_{s,t\in\Omega}$ on $\Omega\times\Omega$ such that $\sum_{s\in\Omega}\rho(s,t)=\sum_{s\in\Omega}\rho(t,s)=q^{-1}$ for all $t\in\Omega$. The function

$$\rho\in\mathcal{R}(\Omega)\mapsto\textstyle\sum_{\sigma,\tau\in\Omega^k}\mathbb{E}[\boldsymbol{\psi}(\sigma)\boldsymbol{\psi}(\tau)]\prod_{i=1}^k\rho(\sigma_i,\tau_i)$$

has the uniform distribution on $\Omega\times\Omega$ as its unique global minimizer.

POS. For all $\pi,\pi'\in\mathcal{P}_*^2(\Omega)$ the following is true. With $\boldsymbol{\rho}_1,\boldsymbol{\rho}_2,\ldots$ chosen from π, $\boldsymbol{\rho}_1',\boldsymbol{\rho}_2',\ldots$ chosen from π' and $\boldsymbol{\psi}\in\Psi$ chosen from P, all mutually independent, we have

$$\begin{aligned}0\quad\leq\quad&\mathbb{E}\left[\Lambda\left(\sum_{\tau\in\Omega^k}\boldsymbol{\psi}(\tau)\prod_{i\in[k]}\boldsymbol{\rho}_i(\tau_i)\right)\right]+(k-1)\mathbb{E}\left[\Lambda\left(\sum_{\tau\in\Omega^k}\boldsymbol{\psi}(\tau)\prod_{i\in[k]}\boldsymbol{\rho}_i'(\tau_i)\right)\right]\\ &-\mathbb{E}\left[k\Lambda\left(\sum_{\tau\in\Omega^k}\boldsymbol{\psi}(\tau)\boldsymbol{\rho}_1(\tau_1)\prod_{i\in[k]\setminus\{1\}}\boldsymbol{\rho}_i'(\tau_i)\right)\right].\end{aligned}\tag{2.3}$$

Conditions similar to **SYM**, **BAL** and **POS** appeared in [13], too. The upshot is that all four conditions can be checked solely by inspecting the distribution P on weight functions, and this is not normally difficult. For a more detailed discussion of these conditions see the full version of this paper in [11].

It is not difficult to cast the Potts antiferromagnet and the k-spin model as factor graph models. For the Potts model we let $k=2$ and we merely introduce a single weight function $\psi_{q,\beta}(\sigma,\tau)=\exp(-\beta\mathbf{1}\{\sigma=\tau\})$. The four conditions **SYM**, **BAL**, **POS** and **MIN** are easily verified. For the k-spin model we need infinitely many weight functions, one for each $J\in\mathbb{R}$, defined by $\psi_{J,\beta}(\sigma_1,\ldots,\sigma_k)=1+\tanh(J\beta)\sigma_1\cdots\sigma_k$, and P is the distribution of $\psi_{\boldsymbol{J},\beta}$ with J a standard Gaussian. The conditions **SYM**, **BAL** and **POS** hold for this model for any k and **MIN** is satisfied for even k.

2.2 Results

We proceed with the results on the condensation phase transition, the limiting distribution of the free energy, the overlap, the reconstruction and the detection thresholds for general random factor graph models.

▶ **Theorem 7.** *Assume that P satisfies **SYM**, **BAL** and **POS** and let $d>0$. With $\boldsymbol{\gamma}$ a $\mathrm{Po}(d)$-random variable, $\boldsymbol{\rho}_1^\pi,\boldsymbol{\rho}_2^\pi,\ldots$ chosen from $\pi\in\mathcal{P}_*^2(\Omega)$ and $\boldsymbol{\psi}_1,\boldsymbol{\psi}_2,\ldots\in\Psi$ chosen from P, all mutually independent, let*

$$\begin{aligned}\mathcal{B}(d,P,\pi) \quad &= \quad \mathbb{E}\left[\tfrac{1}{q\xi^{\boldsymbol{\gamma}}}\Lambda\left(\sum_{\sigma\in\Omega}\prod_{i\in[\boldsymbol{\gamma}]}\sum_{\tau\in\Omega^k}\mathbf{1}\{\tau_k=\sigma\}\boldsymbol{\psi}_i(\tau)\prod_{j\in[k-1]}\boldsymbol{\rho}^\pi_{ki+j}(\tau_j)\right)\right]\\ &\qquad -\tfrac{d(k-1)}{k\xi}\mathbb{E}\left[\Lambda\left(\sum_{\tau\in\Omega^k}\boldsymbol{\psi}_1(\tau)\prod_{i\in[k]}\boldsymbol{\rho}^\pi_j(\tau_j)\right)\right]\end{aligned} \tag{2.4}$$

and let $d_{\mathrm{cond}}=\inf\left\{d>0\,:\,\sup_{\pi\in\mathcal{P}_*^2(\Omega)}\mathcal{B}(d,P,\pi)>\ln q+\frac{d}{k}\ln\xi\right\}$. Then $1/(k-1)\le d_{\mathrm{cond}}<\infty$ and

$$\lim_{n\to\infty}\frac{1}{n}\mathbb{E}[\ln Z(\boldsymbol{G})]\begin{cases}=\ln q+\frac{d}{k}\ln\xi & \text{if } d\le d_{\mathrm{cond}},\\ <\ln q+\frac{d}{k}\ln\xi & \text{if } d>d_{\mathrm{cond}}.\end{cases}$$

Theorem 7 generalizes [13, Theorem 2.7], which requires that the set Ψ of weight functions be finite (and thus does not cover the k-spin model).

Admittedly the formula for d_{cond} provided by Theorem 7 is neither very simple nor very explicit, but we are not aware of any reason why it ought to be. Yet there is a natural generalization of the Kesten-Stigum bound from (1.2) that provides an easy-to-compute upper bound on d_{cond} in terms of the spectrum of a certain linear operator. The operator is constructed as follows. For $\psi\in\Psi$ let $\Phi_\psi\in\mathbb{R}^{\Omega\times\Omega}$ be the matrix with entries

$$\Phi_\psi(\omega,\omega')=q^{1-k}\xi^{-1}\textstyle\sum_{\tau\in\Omega^k}\mathbf{1}\{\tau_1=\omega,\tau_2=\omega'\}\psi(\tau) \qquad (\omega,\omega'\in\Omega) \tag{2.5}$$

and let $\Xi=\Xi_P$ be the linear operator on the q^2-dimensional space $\mathbb{R}^\Omega\otimes\mathbb{R}^\Omega$ defined by

$$\Xi=\Xi_P=\mathbb{E}[\Phi_{\boldsymbol{\psi}}\otimes\Phi_{\boldsymbol{\psi}}]. \tag{2.6}$$

Furthermore, letting $\mathcal{E}=\{z\in\mathbb{R}^q\otimes\mathbb{R}^q:\forall y\in\mathbb{R}^q:\langle z,\mathbf{1}\otimes y\rangle=\langle z,y\otimes\mathbf{1}\rangle=0\}$, with $\mathbf{1}$ denoting the vector with all entries equal to one, we introduce

$$d_{\mathrm{KS}}=\left((k-1)\max_{x\in\mathcal{E}:\|x\|=1}\langle\Xi x,x\rangle\right)^{-1}, \tag{2.7}$$

with the convention that $d_{\mathrm{KS}}=\infty$ if $\max_{x\in\mathcal{E}:\|x\|=1}\langle\Xi x,x\rangle=0$.

▶ **Theorem 8.** *If P satisfies **SYM** and **BAL**, then $d_{\mathrm{cond}} \leq d_{\mathrm{KS}}$.*

We shall see in Section 3 that Ξ is related to the "broadcasting matrix" of a suitable Galton-Watson tree, which justifies referring to d_{KS} as a generalized version of the classical *Kesten-Stigum bound* from [26]. While this bound is not generally tight, it plays a major conceptual role, as will emerge in due course.

Theorem 7 easily implies that $n^{-1}\ln Z(\boldsymbol{G})$ converges to $\ln q + \frac{d}{k}\ln\xi$ in probability if $d < d_{\mathrm{cond}}$. Yet due to the scaling factor of $1/n$ this is but a rough first order approximation. The next theorem, arguably the principal achievement of the paper, yields the exact limiting distribution of the *unscaled* free energy $\ln Z(\boldsymbol{G})$ in the entire replica symmetric phase. Recalling (2.5), let the $\Omega \times \Omega$-matrix

$$\Phi = \Phi_P = \mathbb{E}[\Phi_{\boldsymbol{\psi}}]. \tag{2.8}$$

▶ **Theorem 9.** *Assume that P satisfies **SYM**, **BAL**, **POS** and **MIN** and that $0 < d < d_{\mathrm{cond}}$. Let $(K_l)_{l\geq 1}$ be a family of Poisson variables with means $\mathbb{E}[K_l] = \frac{1}{2l}(d(k-1))^l$ and let $(\boldsymbol{\psi}_{l,i,j})_{l,i,j\geq 1}$ be a sequence of samples from P, all mutually independent. Then the random variable*

$$\mathcal{K} = \sum_{l=1}^{\infty}\left[\frac{(d(k-1))^l}{2l}\left(1-\mathrm{tr}(\Phi^l)\right) + \sum_{i=1}^{K_l}\ln\mathrm{tr}\prod_{j=1}^{l}\Phi_{\boldsymbol{\psi}_{l,i,j}}\right] \tag{2.9}$$

satisfies $\mathbb{E}|\mathcal{K}| < \infty$ and we have the following convergence in distribution:

$$\ln Z(\boldsymbol{G}) - \left(n+\tfrac{1}{2}\right)\ln q - \boldsymbol{m}\ln(\xi) + \tfrac{1}{2}\sum_{\lambda\in\mathrm{Eig}(\Phi)\setminus\{1\}}\ln(1-d(k-1)\lambda) \quad \xrightarrow{n\to\infty} \quad \mathcal{K}. \tag{2.10}$$

Let $\bar\rho$ be the uniform distribution on $\Omega \times \Omega$, while for $\sigma, \tau \in \Omega^{V_n}$ we defined the *overlap* $\rho_{\sigma,\tau}$ such that $\rho_{\sigma,\tau}(\omega,\omega') = |\sigma^{-1}(\omega)\cap\tau^{-1}(\omega')|/n$. The following theorem confirms one of the core tenets of the physicists' cavity method, namely the absence of extensive long-range correlations for $d < d_{\mathrm{cond}}$.

▶ **Theorem 10.** *If **SYM**, **BAL**, **POS** , **MIN** hold, then it holds that*

$$d_{\mathrm{cond}} = \inf\left\{d > 0 : \limsup_{n\to\infty}\mathbb{E}\left\langle\|\rho_{\boldsymbol{\sigma},\boldsymbol{\tau}} - \bar\rho\|_{TV}\right\rangle_{\boldsymbol{G}} > 0\right\}.$$

The condensation phase transition is generally preceded by another threshold where certain point-to-set correlations emerge, the reconstruction threshold [27]. Indeed, the quantity $\mathrm{corr}(d)$ as defined in (1.4) generalises naturally to any random factor graph model. Further, we can easily construct a mulit-type Galton-Watson tree $\boldsymbol{T}(d,P)$ that mimics the local geometry of a random factor graph $\boldsymbol{G}$. Its types are variable and constraint nodes, each of the latter endowed with a weight function $\psi \in \Psi$. The root is a variable node r. The offspring of a variable node is a $\mathrm{Po}(d)$ number of constraint nodes whose weight functions are chosen from P independently. Moreover, the offspring of a constraint node is $k-1$ variable nodes. For an integer $\ell \geq 0$ we let $\boldsymbol{T}^\ell(d,P)$ denote the (finite) tree obtained from $\boldsymbol{T}(d,P)$ by deleting all variable nodes at distance greater than 2ℓ from r. We set

$$\mathrm{corr}^\star(d) = \lim_{\ell\to\infty}\sum_{s\in\Omega}\mathbb{E}\left\langle\left|\left\langle\mathbf{1}\{\boldsymbol{\sigma}(r)=s\}\middle|\nabla_\ell(\boldsymbol{T}^\ell(d,P),r)\right\rangle_{\boldsymbol{T}^\ell(d,P)} - 1/q\right|\right\rangle_{\boldsymbol{T}^\ell(d,P)}. \tag{2.11}$$

The *tree reconstruction threshold* is defined as $d^\star_{\mathrm{rec}} = \inf\{d > 0 : \mathrm{corr}^\star(d) > 0\}$.

▶ **Theorem 11.** *If P satisfies **SYM**, **BAL**, **POS** and **MIN**, then $0 < d_{\mathrm{rec}} = d^\star_{\mathrm{rec}} \leq d_{\mathrm{cond}}$.*

Theorem 11 generalises results from [21, 35]. For further discussion see the full version [11].

Finally, there is a natural statistical inference version of the random factor graph model, the *teacher-student model* [42], a generalisation of the stochastic block model. The model is defined as follows.

TCH1 an assignment $\boldsymbol{\sigma}^* : V_n \to \Omega$, the *ground truth*, is chosen uniformly at random.

TCH2 independently of $\boldsymbol{\sigma}^*$, draw $\boldsymbol{m} = \boldsymbol{m}_d(n)$ from the Poisson distribution with mean dn/k.

TCH3 generate $\boldsymbol{G}^*$ with factor nodes $a_1, \ldots, a_{\boldsymbol{m}}$ by choosing the neighborhoods ∂a_j and the weight functions ψ_{a_j} from the distribution

$$\mathbb{P}\left[\partial a_j = (y_1, \ldots, y_k), \psi_{a_j} \in \mathcal{A}\right] \propto \mathbb{E}[\mathbf{1}\{\boldsymbol{\psi} \in \mathcal{A}\}\boldsymbol{\psi}(\sigma(y_1), \ldots, \sigma(y_k))], \tag{2.12}$$

independently for $i = 1, \ldots, \boldsymbol{m}$.

As in the case of the stochastic block model, the *detection problem* arises: given a factor graph G, for what d is it possible to discern whether G was chosen from the model $\boldsymbol{G}^*$ or from the "null model" $\boldsymbol{G}$? The following theorem shows that the detection threshold is always given by d_{cond}.

▶ **Theorem 12.** *If P satisfies **SYM**, **BAL**, **POS** and **MIN**, then $\boldsymbol{G}, \boldsymbol{G}^*$ are mutually contiguous for all $d < d_{\mathrm{cond}}$, while $\boldsymbol{G}, \boldsymbol{G}^*$ fail to be mutually contiguous for $d > d_{\mathrm{cond}}$.*

The disassortative stochastic block model and the teacher-student model $\boldsymbol{G}^*$ are known to be mutually contiguous [13] and thus Theorem 4 follows from Theorem 12.

3 Proof strategy

The apex of the present work is Theorem 9 about the limiting distribution of the free energy; all the other results follow from it almost immediately. For such a result the usual approach would be the second moment method, pioneered by Achlioptas and Moore [3], in combination with the small subgraph conditioning technique of Robinson and Wormald [25, 41]. However, this approach does not generally allow for tight results (in particular, it typically stops working well below the condensation threshold).

We craft a proof around the teacher-student model $\boldsymbol{G}^*$ instead. Specifically, the main achievement of the recent paper [13] was to verify the cavity formula for the leading order $\lim_{n\to\infty} \frac{1}{n}\mathbb{E}[\ln Z(\boldsymbol{G}^*)]$ of the "free energy" $\ln Z(\boldsymbol{G}^*)$ (in the case that the set Ψ is finite). We will replace the second moment calculation by that free energy formula, generalized to infinite Ψ, and combine it with a suitably generalized small subgraph conditioning technique. The challenge is to integrate these two components seamlessly. We accomplish this by realizing that, remarkably, both arguments are inherently and rather elegantly tied together via the spectrum of the linear operator Ξ from (2.6). But to develop this novel approach we first need to recall the classical second moment argument and understand why it founders.

3.1 Two moments do not suffice

For any second moment calculation it is crucial to fix the number of constraint nodes as otherwise its fluctuations would boost the variance. Hence, we will work with an integer sequence $m = m(n) \geq 0$. We fix $d > 0$ and consider specific integer sequences $m = m(n) \geq 0$ such that $|m(n) - dn/k| \leq n^{3/5}$ for all n. Let $\mathcal{M}(d)$ be the set of all such sequences.

The second moment method rests on showing that $\mathbb{E}[Z(\boldsymbol{G}(n,m))^2] = O(\mathbb{E}[Z(\boldsymbol{G}(n,m))]^2)$. If this is the case, then from Azuma's inequality we get that $\lim_{n\to\infty} n^{-1}\mathbb{E}[\ln Z(\boldsymbol{G}(n,m))] = \lim_{n\to\infty} n^{-1}\ln\mathbb{E}[Z(\boldsymbol{G}(n,m))]$. The second limit is easy to compute because the expectation sits inside the logarithm, and thus we obtain the leading order of the "free energy" $\ln Z(\boldsymbol{G}(n,m))$. In fact, if we can calculate the second moment $\mathbb{E}[Z(\boldsymbol{G}(n,m))^2]$ sufficiently accurately, then it may be possible to determine the limiting distribution of $\ln Z(\boldsymbol{G}(n,m))$ precisely. Suppose that there is a sufficiently simple random variable $Q(\boldsymbol{G}(n,m))$ such that

$$\mathrm{Var}[Z(\boldsymbol{G}(n,m))] = (1+o(1))\mathrm{Var}[\mathbb{E}[Z(\boldsymbol{G}(n,m))|Q(\boldsymbol{G}(n,m))]]. \tag{3.1}$$

The formula

$$\mathrm{Var}[Z(\boldsymbol{G}(n,m))] = \mathrm{Var}[\mathbb{E}[Z(\boldsymbol{G}(n,m))|Q(\boldsymbol{G}(n,m))]] + \mathbb{E}[\mathrm{Var}[Z(\boldsymbol{G}(n,m))|Q(\boldsymbol{G}(n,m))]]$$

implies

$$\mathbb{E}[\mathrm{Var}[Z(\boldsymbol{G}(n,m))|Q(\boldsymbol{G}(n,m))]] = o(\mathbb{E}[Z(\boldsymbol{G}(n,m))]^2) \tag{3.2}$$

and it is not difficult to deduce from (3.2) that $\ln Z(\boldsymbol{G}(n,m)) - \ln\mathbb{E}[Z(\boldsymbol{G}(n,m))|Q(\boldsymbol{G}(n,m))]$ converges to 0 in probability. Hence, we get the limiting distribution of $\ln Z(\boldsymbol{G}(n,m))$ if $Q(\boldsymbol{G}(n,m))$ is simple enough so that the law of $\ln\mathbb{E}[Z(\boldsymbol{G}(n,m))|Q(\boldsymbol{G}(n,m))]$ is easy to express. The basic insight behind the small subgraph conditioning technique is that (3.1) sometimes holds with a variable Q that is determined by the statistics of bounded-length cycles in $\boldsymbol{G}(n,m)$ [25, 41].

Anyhow, the crux of the entire argument is to calculate $\mathbb{E}[Z(\boldsymbol{G}(n,m))^2]$. Stirling's formula yields the following approximation of $\mathbb{E}[Z(\boldsymbol{G}(n,m))^2]$ in terms of the overlaps:

$$\ln\mathbb{E}[Z(\boldsymbol{G}(n,m))^2] = \max_{\rho\in\mathcal{P}(\Omega^2)} n\mathcal{H}(\rho) + m\ln\left(\sum_{s,t\in\Omega^k}\mathbb{E}[\boldsymbol{\psi}(s)\boldsymbol{\psi}(t)]\prod_{i\in[k]}\rho(s_i,t_i)\right) + O(\ln n), \tag{3.3}$$

where $\mathcal{H}(\rho)$ denotes the entropy of ρ. Hence, computing the second moment comes down to identifying the overlap ρ that renders the dominant contribution to the second moment. Indeed, the second moment bound $\mathbb{E}[Z(\boldsymbol{G}(n,m))^2] = O(\mathbb{E}[Z(\boldsymbol{G}(n,m))]^2)$ holds if and only if the maximum (3.3) is attained at the uniform overlap $\bar\rho$. However, this is not generally true for d below but near the condensation threshold.

This problem was noticed and partly remedied in prior work by applying the second moment method to a suitably truncated random variable (e.g. [7, 12]). This method revealed, e.g., the condensation threshold in a few special cases such as the random graph q-coloring problem [7] and the random regular k-SAT model, albeit only for large q and k. Yet apart from introducing such extraneous conditions, arguments of this kind require a meticulous combinatorial study of the specific model.

3.2 The condensation phase transition and the overlap

The merit of the present approach is that we avoid combinatorial deliberations altogether. Instead we employ an asymptotic formula for $\mathbb{E}[\ln Z(\boldsymbol{G}^*)]$ for the teacher-student model $\boldsymbol{G}^*$.

▶ **Theorem 13.** *If P satisfies **SYM**, **BAL** and **POS** and $d > 0$, then with $\mathcal{B}(d,P,\pi)$ from (2.4) we have $\lim_{n\to\infty} n^{-1}\mathbb{E}[\ln Z(\boldsymbol{G}^*)] = \sup_{\pi\in\mathcal{P}_*^2(\Omega)} \mathcal{B}(d,P,\pi)$.*

Theorem 13 was established in [13] for a set Ψ of weight functions that is finite and the proof of Theorem 13 is based on a limiting argument.

We deduce the following result from Theorem 13 by observing that $\frac{\partial}{\partial d}\ln Z(\boldsymbol{G}^*)$ can be expressed in terms of the overlap. Let $\boldsymbol{G}^*(n,m)$ be the teacher-student model with a fixed number m of constraint nodes.

▶ **Proposition 14.** *Assume that* ***BAL, SYM, POS*** *and* ***MIN*** *hold and that $d < d_{\mathrm{cond}}$. There exists a sequence $\zeta = \zeta(n)$, $\zeta(n) = o(1)$ but $n^{1/6}\zeta(n) \to \infty$ as $n \to \infty$, such that for all $m \in \mathcal{M}(d)$ we have*

$$\mathbb{E}\left\langle \|\rho_{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \bar{\rho}\|_{\mathrm{TV}}\right\rangle_{\boldsymbol{G}^*(n,m)} \le \zeta^2. \tag{3.4}$$

Proposition 14 resolves our second moment troubles. Indeed, it enables a generic way of setting up a 'truncated' random variable: with ζ from Proposition 14 we define

$$\mathcal{Z}(G) = Z(G)\mathbf{1}\left\{\left\langle \|\rho_{\boldsymbol{\sigma}_1,\boldsymbol{\sigma}_2} - \bar{\rho}\|_{\mathrm{TV}}\right\rangle_G \le \zeta\right\}. \tag{3.5}$$

Hence, $\mathcal{Z}(G) = Z(G)$ if "most" pairs $\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2$ drawn from μ_G have overlap close to $\bar{\rho}$, and $\mathcal{Z}(G) = 0$ otherwise. Since up to contiguity the teacher-student model $\boldsymbol{G}^*(n,m)$ corresponds to a reweighted version of the random factor graph model $\boldsymbol{G}(n,m)$ where each graph G is weighted according to its partition function $Z(G)$, Proposition 14 shows immediately that this truncation does not diminish the first moment.

▶ **Corollary 15.** *If* ***BAL, SYM, POS*** *and* ***MIN*** *hold and $d < d_{\mathrm{cond}}$, then $\mathbb{E}[\mathcal{Z}(\boldsymbol{G}(n,m))] \sim \mathbb{E}[Z(\boldsymbol{G}(n,m))]$ uniformly for all $m \in \mathcal{M}(d)$.*

The second moment calculation for $\mathcal{Z}$ is easy, too. Indeed, the very construction (3.5) of $\mathcal{Z}$ guarantees that the dominant contribution to the second moment of $\mathcal{Z}$ comes from pairs with an overlap close to $\bar{\rho}$. Hence, computing the second moment comes down to expanding the right hand side of (3.3) around $\bar{\rho}$ via the Laplace method. Yet in order to do so we need to verify that $\bar{\rho}$ is a local maximum of the function

$$\rho \in \mathcal{P}(\Omega^2) \mapsto \mathcal{H}(\rho) + \frac{d}{k}\ln \sum_{s,t\in\Omega^k} \mathbb{E}[\boldsymbol{\psi}(s)\boldsymbol{\psi}(t)]\prod_{i=1}^{k}\rho(s_i,t_i) \tag{3.6}$$

from (3.3). For the special case of the Potts antiferromagnet the overlap concentration (3.4) was established and the second moment argument for $\mathcal{Z}$ was carried out in [13]. While the generalization to random factor graph models is anything but straightforward, an even more important difference lies in the application of the Laplace method. But of course there ought to be a general, conceptual explanation. As we shall see momentarily, there is one indeed, namely the generalized Kesten-Stigum bound.

3.3 The Kesten-Stigum bound

To see the connection, we observe that the Hessian of (3.6) at the point $\bar{\rho}$ is equal to $q(\mathrm{id} - d(k-1)\Xi)$, where Ξ us the matrix from (2.6). Hence, taking into account that the argument ρ is a probability distribution on $\Omega \times \Omega$, we find that $\bar{\rho}$ is a local maximum of (3.6) if and only if

$$\langle(\mathrm{id} - d(k-1)\Xi)x, x\rangle > 0 \qquad \text{for all } x \in \mathbb{R}^q \otimes \mathbb{R}^q \text{ such that } x \perp \mathbf{1}\otimes\mathbf{1}. \tag{3.7}$$

In order to get a handle on the spectrum of the operator Ξ from (2.6) we begin with the following observation about the matrices Φ_ψ and Φ from (2.5) and (2.8).

▶ **Lemma 16.** *Let P satisfy* $\boldsymbol{SYM}$*. Then the matrix Φ_ψ is stochastic and thus $\Phi_\psi\mathbf{1} = \mathbf{1}$ for every $\psi \in \Psi$. Moreover, Φ is symmetric and doubly-stochastic. If, additionally, P satisfies* $\boldsymbol{BAL}$*, then* $\max_{x\perp\mathbf{1}} \langle \Phi x, x\rangle \le 0$.

Proceeding to the operator Ξ, we recall the definition of $\mathcal{E}$ from (2.7) and we introduce

$$\mathcal{E}' = \{x \in \mathbb{R}^q \otimes \mathbb{R}^q : \langle x, \mathbf{1}\otimes\mathbf{1}\rangle = 0\} \supset \mathcal{E}. \tag{3.8}$$

▶ **Lemma 17.** *Assume that P satisfies* $\boldsymbol{SYM}$, $\boldsymbol{BAL}$*. The operator Ξ is self-adjoint, $\Xi(\mathbf{1}\otimes\mathbf{1}) = \mathbf{1}\otimes\mathbf{1}$ and for every $x \in \mathbb{R}^q$ we have $\Xi(x\otimes\mathbf{1}) = (\Phi x)\otimes\mathbf{1}$, $\Xi(\mathbf{1}\otimes x) = \mathbf{1}\otimes(\Phi x)$ and*

$$\langle \Xi(x\otimes\mathbf{1}), x\otimes\mathbf{1}\rangle \le 0, \qquad \langle \Xi(\mathbf{1}\otimes x), \mathbf{1}\otimes x\rangle \le 0 \qquad \text{if } x \perp \mathbf{1}. \tag{3.9}$$

Furthermore, $\Xi\mathcal{E} \subset \mathcal{E}$ and $\Xi\mathcal{E}' \subset \mathcal{E}'$.

Lemma 17 shows that Ξ induces a self-adjoint operator on the space $\mathcal{E}$.

The following proposition yields a bound on the spectral radius of this operator. Let $\mathrm{Eig}^*(\Xi) = \{\lambda \in \mathbb{R} : \exists x \in \mathcal{E}\setminus\{0\} : \Xi x = \lambda x\}$.

▶ **Proposition 18.** *If P satisfies* $\boldsymbol{SYM}$ *and* $\boldsymbol{BAL}$*, then* $d_{\mathrm{cond}}(k-1)\max_{\lambda\in\mathrm{Eig}^*(\Xi)}|\lambda| \le 1$.

The proof of Proposition 18 is based on establishing an inherent connection between the spectrum of Ξ and the Bethe free energy functional $\mathcal{B}$ from (2.4). Specifically, we use the eigenvector of Ξ to construct a candidate maximum of the functional $\mathcal{B}$. Theorem 8 is immediate from Proposition 18.

Lemma 17 and Proposition 18 show that (3.7) is satisfied, and thus that $\bar\rho$ is a local maximum of (3.6), for all $d < d_{\mathrm{cond}}$. Indeed, it is immediate from (3.9) that $\langle(\mathrm{id} - d(k-1)\Xi)x, x\rangle > 0$ if x is of the form $\mathbf{1}\otimes y$ or $y\otimes\mathbf{1}$ for some $\mathbf{1} \perp y \in \mathbb{R}^q$, and Theorem 8 shows that $\langle(\mathrm{id} - d(k-1)\Xi)x, x\rangle > 0$ for all $x \in \mathcal{E}$. Hence, Proposition 18 links the free energy calculation for $\boldsymbol{G}^*$ with the second moment of $\mathcal{Z}$.

3.4 Second moment redux

Observe that by Lemma 16 the set $\mathrm{Eig}(\Phi)$ of eigenvalues of Φ contains precisely one nonnegative element, namely 1. Therefore, the following formula makes sense.

▶ **Proposition 19.** *Suppose that P satisfies* $\boldsymbol{SYM}$ *and* $\boldsymbol{BAL}$ *and let $0 < d$. Then uniformly for all $m \in \mathcal{M}(d)$,*

$$\mathbb{E}[Z(\boldsymbol{G}(n,m))] \sim \frac{q^{n+\frac{1}{2}}\xi^m}{\prod_{\lambda\in\mathrm{Eig}(\Phi)\setminus\{1\}}\sqrt{1-d(k-1)\lambda}}. \tag{3.10}$$

Proceeding to the second moment, we recall from Lemma 17 that Ξ induces an endomorphism on the subspace $\mathcal{E}'$ from (3.8) and for the spectrum of Ξ on $\mathcal{E}'$ we write

$$\mathrm{Eig}'(\Xi) = \{\lambda \in \mathbb{R} : \exists x \in \mathcal{E}'\setminus\{0\} : \Xi x = \lambda x\}.$$

Lemma 17 and Proposition 18 imply that $d_{\mathrm{cond}}(k-1)\lambda \le 1$ for all $\lambda \in \mathrm{Eig}'(\Xi)$. Therefore, the following formula for the second moment makes sense, too.

▶ **Proposition 20.** *If P satisfies* $\boldsymbol{SYM}$ *and* $\boldsymbol{BAL}$ *and let $0 < d < d_{\mathrm{cond}}$. Then uniformly for all $m \in \mathcal{M}(d)$,*

$$\mathbb{E}[\mathcal{Z}(\boldsymbol{G}(n,m))^2] \le \frac{(1+o(1))q^{2n+1}\xi^{2m}}{\prod_{\lambda\in\mathrm{Eig}'(\Xi)}\sqrt{1-d(k-1)\lambda}}. \tag{3.11}$$

Combining Corollary 15 with Propositions 19 and 20 and applying Lemma 17, we obtain for $m \in \mathcal{M}(d)$,

$$\frac{\mathbb{E}[\mathcal{Z}(\boldsymbol{G}(n,m))^2]}{\mathbb{E}[\mathcal{Z}(\boldsymbol{G}(n,m))]^2} \sim \frac{\prod_{\lambda \in \mathrm{Eig}(\Phi)\setminus\{1\}} 1-d(k-1)\lambda}{\prod_{\lambda \in \mathrm{Eig}'(\Xi)} \sqrt{1-d(k-1)\lambda}} = \prod_{\lambda \in \mathrm{Eig}^*(\Xi)} \frac{1}{\sqrt{1-d(k-1)\lambda}} \qquad \text{if } d < d_{\mathrm{cond}}. \tag{3.12}$$

In particular, the ratio of the second moment and the square of the first is bounded as $n \to \infty$.

3.5 Virtuous cycles

In order to determine the limiting distribution of $\ln Z(\boldsymbol{G}(n,m))$ we are going to "explain" the remaining variance of $\mathcal{Z}(\boldsymbol{G}(n,m))$ in terms of the statistics of the bounded-length cycles of $\boldsymbol{G}(n,m)$. However, by comparison to prior applications of the small subgraph conditioning technique, here it does not suffice to merely record how many cycles of a given length occur. We also need to take into account the specific weight functions along the cycle. Yet this approach is complicated substantially by the fact that there may be infinitely many different weight functions. To deal with this issue we are going to discretize the set of weight functions and perform a somewhat delicate limiting argument.

For integer $\ell > 0$, $E_1, \ldots, E_\ell \subset \Psi$ and $s_1, t_1, \ldots, s_\ell, t_\ell \in \{1, \ldots, k\}$ a *signature of order ℓ* is a family

$$Y = (E_1, s_1, t_1, E_2, s_2, t_2, \ldots, E_\ell, s_\ell, t_\ell)$$

such that $s_i \neq t_i$ for all $i \in \{1, \ldots, \ell\}$ and $s_1 < t_1$ if $\ell = 1$. We let $\mathcal{Y}$ be the set of all signatures.

For a factor graph G we call a family $(x_{i_1}, a_{h_1}, \ldots, x_{i_\ell}, a_{h_\ell})$ a *cycle of signature Y in G* if the following holds: All $i_1, \ldots, i_\ell \in \{1, \ldots, n\}$ are pairwise distinct, the same holds for $h_1, \ldots, h_\ell \in \{1, \ldots, m\}$. We impose an orientation on how we traverse the cycle, i.e. we start from x_{i_1} and we traverse towards the constraint node with the smaller index or $s_1 < t_1$ if $\ell = 1$. For this reason we require $i_1 = \min\{i_1, \ldots, i_\ell\}$, while $h_1 < h_\ell$ if $\ell > 1$. The weight functions along the cycle belong to $E_1, \ldots, E_\ell$, i.e. $\psi_{a_{h_j}} \in E_j$, for $j = 1, \ldots, \ell$. Finally, we require that the cycle enters the jth constraint node in position s_j and leaves in position t_j.

Let $C_Y(G)$ denote the number of cycles of signature Y. Moreover, for an event $\mathcal{A} \subset \Psi$ with $P(\mathcal{A}) > 0$ and $h, h' \in \{1, \ldots, k\}$ define the $q \times q$ matrix $\Phi_{\mathcal{A},h,h'}$ by letting

$$\Phi_{\mathcal{A},h,h'}(\omega,\omega') = q^{1-k}\xi^{-1} \sum_{\tau \in \Omega^k} \mathbf{1}\{\tau_h = \omega, \tau_{h'} = \omega'\} \mathbb{E}[\boldsymbol{\psi}(\tau)|\mathcal{A}] \qquad (\omega, \omega' \in \Omega). \tag{3.13}$$

In addition, for a signature $Y = (E_1, s_1, t_1, \ldots, E_\ell, s_\ell, t_\ell)$ define

$$\kappa_Y = \frac{1}{2\ell}\left(\frac{d}{k}\right)^\ell \prod_{i=1}^{\ell} P(E_i), \qquad \Phi_Y = \prod_{i=1}^{\ell} \Phi_{E_i,s_i,t_i}, \qquad \hat{\kappa}_Y = \kappa_Y \operatorname{tr}(\Phi_Y). \tag{3.14}$$

A *cycle of order ℓ* is a family $(x_{i_1}, a_{h_1}, \ldots, x_{i_\ell}, a_{h_\ell})$ of signature $(\Psi, s_1, t_1, \ldots, \Psi, s_\ell, t_\ell)$ for some sequence $s_1, t_1, \ldots, s_\ell, t_\ell$, and we let C_ℓ signify the number of such cycles. Finally, two signatures $Y = (E_1, s_1, t_1, \ldots, E_\ell, s_\ell, t_\ell)$, $Y' = (E'_1, s'_1, t'_1, \ldots, E'_{\ell'}, s'_{\ell'}, t'_{\ell'})$ are *disjoint* if either $\ell \neq \ell'$, or for some for some i we have $(s_i, t_i) \neq (s'_i, t'_i)$ or $E_i \cap E'_i = \emptyset$. We establish the following enhancement that takes the weight functions along the cycles into account.

▶ **Proposition 21.** *Suppose that P satisfies* ***SYM*** *and* ***BAL****. Let $Y_1, Y_2, \dots Y_l \in \mathcal{Y}$ be pairwise disjoint signatures and let $y_1, \dots, y_l$ be non-negative integers. Let $d > 0$. Then uniformly for all $m \in \mathcal{M}(d)$,*

$$\mathbb{P}\left[\forall t \le l:\ C_{Y_t}(\boldsymbol{G}(n,m)) = y_t\right] \sim \prod_{t=1}^{l} \mathbb{P}\left[\mathrm{Po}(\kappa_{Y_t}) = y_t\right],$$
$$\mathbb{P}\left[\forall t \le l:\ C_{Y_t}(\boldsymbol{G}^*(n,m)) = y_t\right] \sim \prod_{t=1}^{l} \mathbb{P}\left[\mathrm{Po}(\hat\kappa_{Y_t}) = y_t\right].$$

Thus, for disjoint $Y_1, \dots, Y_l$ the cycle counts C_{Y_t} are asymptotically independent Poisson.

Finally, we establish that $\mathcal{K}$ from Theorem 9 is well-defined. We view $\Psi \subset [0,2]^{\Omega^k}$ as a subset of a cube in Euclidean speace. For an integer $r \ge 1$ let $\mathfrak{C}_r$ be the partition of Ψ induced by slicing the cube into pairwise disjoint sub-cubes of side length $1/r$. Further, let $\mathcal{Y}_{\ell,r}$ denote the set of all signatures $(E_1, s_1, t_1, \dots, E_\ell, s_\ell, t_\ell)$ such that $E_1, \dots, E_\ell \in \mathfrak{C}_r$ and such that $P(E_i) > 0$ for all $i \le \ell$, and define $\mathcal{Y}_{\le \ell,r} = \bigcup_{l=1}^{\ell} \mathcal{Y}_{l,r}$. Furthermore, if $\psi \in \Psi$ belongs to a sub-cube $C \in \mathfrak{C}_r$, then we let

$$\psi^{(r)}(\tau) = \mathbb{E}[\boldsymbol{\psi}(\tau)|C] \qquad\qquad (\tau \in \Omega^k).$$

▶ **Proposition 22.** *Assume that P satisfies* ***SYM*** *and* ***BAL*** *and let $0 < d < d_{\mathrm{cond}}$. Let $(K_l)_{l \ge 1}$ be a family of independent Poisson variables with $\mathbb{E}[K_l] = (d(k-1))^l/(2l)$ and let $(\boldsymbol{\psi}_{l,i,j})_{l,i,j}$ be a family of independent samples from P. Furthermore, define*

$$\mathcal{K}_{\ell,r} = \sum_{l=1}^{\ell}\left[\frac{(d(k-1))^l}{2l}\left(1 - \mathrm{tr}(\Phi^l)\right) + \sum_{i=1}^{K_l} \ln \mathrm{tr} \prod_{j=1}^{l} \Phi_{\boldsymbol{\psi}^{(r)}_{l,i,j}}\right],$$
$$\mathcal{K}_{\ell} = \sum_{l=1}^{\ell}\left[\frac{(d(k-1))^l}{2l}\left(1 - \mathrm{tr}(\Phi^l)\right) + \sum_{i=1}^{K_l} \ln \mathrm{tr} \prod_{j=1}^{l} \Phi_{\boldsymbol{\psi}_{l,i,j}}\right]$$

and $\mathcal{K} = \sum_{\ell=1}^{\infty} \mathcal{K}_\ell$. Then all $\mathcal{K}_{\ell,r}$ are uniformly bounded in the L^1-norm, $\mathcal{K}_{\ell,r}$ is L^1-convergent to $\mathcal{K}_\ell$ as $r \to \infty$ and $\mathcal{K}_\ell$ is L^1-convergent to $\mathcal{K}$ as $\ell \to \infty$. Furthermore,

$$\lim_{\ell\to\infty}\lim_{r\to\infty} \exp \sum_{Y \in \mathcal{Y}_{\le \ell,r}} \frac{(\kappa_Y - \hat\kappa_Y)^2}{\kappa_Y} = \prod_{\lambda \in \mathrm{Eig}^*(\Xi)} \frac{1}{\sqrt{1 - d(k-1)\lambda}}.$$

Equipped with Propositions 19–22 we can determine the limiting distribution of $\ln Z(\boldsymbol{G})$ and thus prove Theorem 9 by applying Janson's version of the small subgraph conditioning theorem [25] if the set Ψ is finite. In the case of infinite Ψ additional steps are necessary, see in the full version of this paper in [11].

Acknowledgment. We thank Will Perkins, Guilhem Semerjian and Nick Wormald for helpful discussions.

References

1 Emmanuel Abbe. Community detection and stochastic block models: recent developments. *CoRR*, abs/1703.10146, 2017. URL: `http://arxiv.org/abs/1703.10146`.

2 Emmanuel Abbe and Colin Sandon. Detection in the stochastic block model with multiple clusters: proof of the achievability conjectures, acyclic bp, and the information-computation gap. *CoRR*, abs/1512.09080, 2015. URL: `http://arxiv.org/abs/1512.09080`.

3 Dimitris Achlioptas and Cristopher Moore. Random k-SAT: Two moments suffice to cross a sharp threshold. *SIAM J. Comput.*, 36(3):740–762, 2006.

4 Dimitris Achlioptas, Assaf Naor, and Yuval Peres. Rigorous location of phase transitions in hard optimization problems. *Nature*, 435(7043):759–764, 06 2005.

5 Jess Banks, Cristopher Moore, Joe Neeman, and Praneeth Netrapalli. Information-theoretic thresholds for community detection in sparse networks. In *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, pages 383–416, 2016.

6 Victor Bapst and Amin Coja-Oghlan. Harnessing the bethe free energy. *Random Struct. Algorithms*, 49(4):694–741, 2016.

7 Victor Bapst, Amin Coja-Oghlan, Samuel Hetterich, Felicia Raßmann, and Dan Vilenchik. The condensation phase transition in random graph coloring. *Communications in Mathematical Physics*, 341(2):543–606, 2016.

8 Jean Barbier, Mohamad Dia, Nicolas Macris, Florent Krzakala, Thibault Lesieur, and Lenka Zdeborová. Mutual information for symmetric rank-one matrix estimation: A proof of the replica formula. In *Advances in Neural Information Processing Systems 29*, pages 424–432. Curran Associates, Inc., 2016.

9 Mohsen Bayati, David Gamarnik, and Prasad Tetali. Combinatorial approach to the interpolation method and scaling limits in sparse random graphs. *Ann. Probab.*, 41(6):4080–4115, 11 2013.

10 Amin Coja-Oghlan. Phase transitions in discrete structures. In *7th European Congress of Mathematics*, (In press) 2016.

11 Amin Coja-Oghlan, Charilaos Efthymiou, Nor Jaafari, Mihyun Kang, and Tobias Kapetanopoulos. Charting the replica symmetric phase. *CoRR*, abs/1704.01043, 2017. URL: `https://arxiv.org/abs/1704.01043`.

12 Amin Coja-Oghlan and Nor Jaafari. On the potts antiferromagnet on random graphs. *Electr. J. Comb.*, 23(4):P4.3, 2016.

13 Amin Coja-Oghlan, Florent Krzakala, Will Perkins, and Lenka Zdeborová. Information-theoretic thresholds from the cavity method. *CoRR*, abs/1611.00814, 2016. URL: `http://arxiv.org/abs/1611.00814`.

14 Pierluigi Contucci, Sander Dommers, Cristian Giardinà, and Shannon Starr. Antiferromagnetic potts model on the Erdős-Rényi random graph. *Communications in Mathematical Physics*, 323(2):517–554, 2013.

15 Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106–, 12 2011.

16 Yash Deshpande, Emmanuel Abbe, and Andrea Montanari. Asymptotic mutual information for the binary stochastic block model. In *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*, pages 185–189, 2016.

17 Jian Ding, Allan Sly, and Nike Sun. Proof of the satisfiability conjecture for large k. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 59–68, 2015.

18 Ulisse Ferrari, Carlo Lucibello, Flaviano Morone, Giorgio Parisi, Federico Ricci-Tersenghi, and Tommaso Rizzo. Finite-size corrections to disordered systems on Erdős-Rényi random graphs. *Physical Review B*, 88(18):184201–, 11 2013.

19 Silvio Franz, Michele Leone, Federico Ricci-Tersenghi, and Riccardo Zecchina. Exact solutions for diluted spin glasses and optimization problems. *Physical Review Letters*, 87(12:127209), 08 2001.

20 Andreas Galanis, Daniel Stefankovic, and Eric Vigoda. Inapproximability for antiferromagnetic spin systems in the tree nonuniqueness region. *J. ACM*, 62(6):50:1–50:60, 2015.

21 Antoine Gerschenfeld and Andrea Montanari. Reconstruction for models on random graphs. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), October 20-23, 2007, Providence, RI, USA, Proceedings*, pages 194–204, 2007.

22 Andrei Giurgiu, Nicolas Macris, and Rüdiger L. Urbanke. Spatial coupling as a proof technique and three applications. *IEEE Trans. Information Theory*, 62(10):5281–5295, 2016.

23 Francesco Guerra and Fabio Lucio Toninelli. The high temperature region of the Viana–Bray diluted spin glass model. *Journal of Statistical Physics*, 115(1):531–555, 2004.

24 Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

25 Svante Janson. Random regular graphs: Asymptotic distributions and contiguity. *Combinatorics, Probability & Computing*, 4:369–405, 1995.

26 Harry Kesten and Bernt P. Stigum. Additional limit theorems for indecomposable multidimensional galton-watson processes. *Ann. Math. Statist.*, 37(6):1463–1481, 1966. `doi:10.1214/aoms/1177699139`.

27 Florent Krzakała, Andrea Montanari, Federico Ricci-Tersenghi, Guilhem Semerjian, and Lenka Zdeborová. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proceedings of the National Academy of Sciences*, 104(25):10318–10323, 06 2007.

28 Marc Lelarge and Léo Miolane. Fundamental limits of symmetric low-rank matrix estimation. *CoRR*, abs/1611.03888, 2016. URL: `https://arxiv.org/abs/1611.03888`.

29 Carlo Lucibello, Flaviano Morone, Giorgio Parisi, Federico Ricci-Tersenghi, and Tommaso Rizzo. Finite-size corrections to disordered ising models on random regular graphs. *Physical Review E*, 90(1):012146–, 07 2014.

30 Laurent Massoulié. Community detection thresholds and the weak ramanujan property. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 694–703, 2014.

31 Marc Mézard and Andrea Montanari. *Information, physics and computation.* Oxford University Press, 2009.

32 Marc Mézard and Giorgio Parisi. The bethe lattice spin glass revisited. *Eur. Phys. J. B*, 20(2):217–233, 3 2001.

33 Marc Mézard, Giorgio Parisi, and Ricardo Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297(5582):812, 08 2002.

34 Marc Mézard, Federico Ricci-Tersenghi, and Riccardo Zecchina. Two solutions to diluted p-spin models and XORSAT problems. *Journal of Statistical Physics*, 111(3):505–533, 2003.

35 Andrea Montanari, Ricardo Restrepo, and Prasad Tetali. Reconstruction and clustering in random constraint satisfaction problems. *SIAM J. Discrete Math.*, 25(2):771–808, 2011.

36 Cristopher Moore. The computer science and physics of community detection: Landscapes, phase transitions, and hardness. *CoRR*, abs/1702.00467, 2017. URL: `http://arxiv.org/abs/1702.00467`.

37 Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *CoRR*, abs/1311.4115, 2013. URL: `http://arxiv.org/abs/1311.4115`.

38 Elchanan Mossel, Joe Neeman, and Allan Sly. Reconstruction and estimation in the planted partition model. *Probability Theory and Related Fields*, 162(3):431–461, 2015. `doi:10.1007/s00440-014-0576-6`.

39 Paul Erdős and Alfred Rényi. On the evolution of random graphs. *Magyar Tud. Akad. Mat. Kutató Int. Közl*, 5:17–61, 1960.

40 Tom Richardson and Rüdiger Urbanke. *Modern coding theory.* Cambridge University Press, 2008.

41 Robert W. Robinson and Nicholas C. Wormald. Almost all cubic graphs are hamiltonian. *Random Struct. Algorithms*, 3(2):117–126, 1992.

42 Lenka Zdeborová and Florent Krzakala. Statistical physics of inference: thresholds and algorithms. *Advances in Physics*, 65(5):453–552, 2016.

Probabilistic Logarithmic-Space Algorithms for Laplacian Solvers

Dean Doron[*1], **François Le Gall**[†2], **and Amnon Ta-Shma**[‡1]

1 **The Blavatnik School of Computer Science, Tel-Aviv University, Tel Aviv, Israel**
deandoron@mail.tau.ac.il, amnon@tau.ac.il

2 **Graduate School of Informatics, Kyoto University, Kyoto, Japan**
legall@i.kyoto-u.ac.jp

Abstract

A recent series of breakthroughs initiated by Spielman and Teng culminated in the construction of nearly linear time Laplacian solvers, approximating the solution of a linear system $\mathcal{L}x = b$, where $\mathcal{L}$ is the normalized Laplacian of an undirected graph. In this paper we study the *space complexity* of the problem. Surprisingly we are able to show a probabilistic, *logspace* algorithm solving the problem. We further extend the algorithm to other families of graphs like Eulerian graphs (and directed regular graphs) and graphs that mix in polynomial time.

Our approach is to pseudo-invert the Laplacian, by first "peeling-off" the problematic kernel of the operator, and then to approximate the inverse of the remaining part by using a Taylor series. We approximate the Taylor series using a previous work and the special structure of the problem. For directed graphs we exploit in the analysis the Jordan normal form and results from matrix functions.

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems

Keywords and phrases Laplacian solvers, Randomized logspace, Bounded-space complexity classes, Random walks, Matrix computation

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.41

1 Introduction

Approximating the solution of a linear system $\mathcal{L}x = b$, where $\mathcal{L}$ is the normalized *Laplacian* of a graph G, is an important algorithmic challenge with multitude of algorithmic applications (see [39] and references therein). In the time-bounded setting this problem has drawn a lot of attention over the past decade. A series of breakthroughs initiated by Spielman and Teng culminated in the construction of almost linear-time algorithms [24, 29, 33, 34, 35, 36].

We are interested in studying the *space* complexity of this problem, and specifically achieving a probabilistic logspace algorithm that approximates a solution to such a system. We show that the class BPL is powerful enough to approximate the solution to a linear

[*] Supported by the Israel science Foundation grant no. 994/14 and by the United States – Israel Binational Science Foundation grant no. 2010120. This work was done in part while the author was visiting Kyoto University.

[†] Supported by the Grant-in-Aid for Young Scientists (A) No. 16H05853, the Grant-in-Aid for Scientific Research (A) No. 16H01705, and the Grant-in-Aid for Scientific Research on Innovative Areas No. 24106009 of the Japan Society for the Promotion of Science and the Ministry of Education, Culture, Sports, Science and Technology in Japan.

[‡] Supported by the Israel science Foundation grant no. 994/14 and by the United States – Israel Binational Science Foundation grant no. 2010120.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 41; pp. 41:1–41:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

system of equations for a wide and important variety of linear operators, and in particular for Laplacians of undirected graph (which is the focus of the work of Spielman and Teng). In fact we do more and approximate a *generalized inverse* of the Laplacian, i.e., a matrix $\mathcal{L}^\star$ such that $\mathcal{L}\mathcal{L}^\star\mathcal{L} = \mathcal{L}$, which is sufficient for solving such a set of equations. In essence this means that we invert the matrix on the subspace defined by its image, leaving the kernel unchanged. We prove:

▶ **Theorem 1.** *There exists a probabilistic algorithm that gets as input an $n \times n$ stochastic matrix $\mathcal{S}$ that is the transition matrix of an undirected graph and desired accuracy and confidence parameters $\varepsilon, \delta > 0$, and outputs with probability at least $1 - \delta$ an approximation of the generalized inverse $\mathcal{L}^\star = (\mathcal{I} - \mathcal{S})^\star$ to within an ε-accuracy, using*

$$O\left(\log \frac{n}{\varepsilon} + \log\log \frac{1}{\delta}\right)$$

space.

We are not aware of any previous space bounded algorithm approximating the solution of Laplacian systems.

It is commonly believed that $\mathsf{BPL} = \mathsf{L}$.[1] There are not too many natural, non-trivial problems in L, with the exception of undirected st-connectivity (and the problems that reduce to it [26, 2]) that was solved by Reingold with an intricate and beautiful algorithm [30] (see also [38]). The situation is similar with BPL. Thus the fact that probabilistic logspace algorithms are capable of approximating a solution to a large class of linear-algebra problems comes as a surprise.[2]

We now proceed to discuss our technique. Our goal is to approximate $f(\mathcal{S})$ where f is the function corresponding to the generalized inverse of $\mathcal{I} - \mathcal{S}$. We begin by considering the simpler case where f has a Taylor expansion.

Let G be a regular undirected graph with an associated transition matrix $\mathcal{S}$. As G is undirected and regular, $\mathcal{S}$ is normal and we can represent it as $\mathcal{S} = V\mathbf{\Sigma}V^\dagger$ where $\mathbf{\Sigma}$ is a diagonal matrix with the eigenvalues of $\mathcal{S}$ lying on the diagonal. Consider a function f with a Taylor expansion $f(x) = \sum_i c_i x^i$. We would like to approximate $f(\mathcal{S}) = \sum_i c_i \mathcal{S}^i = Vf(\mathbf{\Sigma})V^\dagger$.[3] Using Taylor expansion in the space-bounded setting is appealing, as in BPL we can approximate powers of stochastic matrices (in fact, even matrices with induced ℓ_∞ norm of at most 1 [17]). Hence, if the series expansion of f behaves "nicely", we can also approximate $f(\mathcal{S})$ in BPL. Using this approach we can, e.g., approximate the matrix $e^{\mathcal{S}}$ using the Taylor expansion $e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!}$.

We now consider the real problem which is approximating a generalized inverse of the Laplacian $\mathcal{L} = \mathcal{I} - \mathcal{S}$. This means that we want to invert $\mathcal{L} = \mathcal{I} - \mathcal{S}$ on its image, leaving the kernel unchanged. Thus, the function f we want to compute is $\frac{1}{1-x}$ when $x \neq 1$ and 1 otherwise (think of x here as an eigenvalue of $\mathcal{S}$). The function f is not continuous and

[1] Some support for this conjecture is given by the following results. Nisan [27] constructed a pseudorandom generator against logspace-bounded non-uniform algorithms that uses seed length $O(\log^2 n)$. Using that he showed BPL is contained in the class having simultaneously polynomial time and $O(\log^2 n)$ space [28]. Saks and Zhou [31] showed that BPL is contained in $\mathsf{DSPACE}(\log^{1.5} n)$. Reingold [30] showed undirected st-connectivity can be solved in deterministic logspace. $\mathsf{BPL} = \mathsf{L}$ is also implied by the conjectured existence of certain circuit lower bounds [25].

[2] Note that finding the *exact* inverse of a matrix, as well as many other important problems in linear algebra, is *complete* for the class $\mathsf{DET} \subseteq \mathsf{NC}^2$ – the class of languages that are NC^1 Turing-reducible to computing the determinant of an integer matrix [5, 7, 14].

[3] The fact that $\sum_i c_i \mathcal{S}^i = Vf(\mathbf{\Sigma})V^\dagger$ is a theorem, see, e.g., [23].

so does not have a Taylor series around 1. Also notice that the operator $\mathcal{L}$ always has a non-trivial kernel (1 is always an eigenvalue of $\mathcal{S}$). Thus, we cannot directly employ the Taylor series approach.

Our solution to the problem is to first "peel-off" the 1-eigenspace using the stationary distribution of the corresponding random walk on G. We are then left with an invertible operator $\mathcal{I} - \mathcal{A}$ whose eigenvalues are bounded away from 0. We now wish to use the Taylor series approach and approximate $(\mathcal{I} - \mathcal{A})^{-1}$ by $\sum_{i=0}^{\infty} \mathcal{A}^i$, which corresponds to the Taylor series $\frac{1}{1-x} = \sum_{i=0}^{\infty} x^i$. There is yet one obstacle we need to overcome, which is that the operator $\mathcal{A}$ that we get after peeling off the stationary distribution of G, is *not* stochastic, and in fact has ℓ_∞ norm larger than 1. Thus, offhand, we do not necessarily know how to simulate high powers of it in BPL. Nevertheless, we exploit its unique structure and show it can be simulated in BPL. Finally, by recovering the peeled-off layer, we essentially recover the required operator $\mathcal{L}^\star$.

We now take a step further, and consider *directed* graphs. The directed case poses major challenges, even if just for the mere fact that directed graphs are not necessarily diagonalizable. In fact, even directed graphs with a favorable structure such as vertex-transitive graphs can be non diagonalizable [20]. The directed Laplacian and its application were studied in, e.g., [6, 12, 3]. Recently, Cohen et al. [13] gave faster algorithms for computing fundamental quantities associated with random walks on directed graphs by improving the running time of solving directed Laplacian systems.

Any operator $\mathcal{A}$ can be represented by its singular value decomposition (SVD) $\mathcal{A} = U\mathbf{\Sigma}V$, where U and V are unitary, and $\mathbf{\Sigma}$ is diagonal with the singular values on the diagonal. Another representation of $\mathcal{A}$ is by its *Jordan normal form*, $\mathcal{A} = V\mathbf{A}V^{-1}$, where V is a basis and $\mathbf{A}$ is the matrix of Jordan blocks. The elements on the diagonals of the Jordan blocks are the eigenvalues of $\mathcal{A}$ (with multiplicity as the multiplicity of the roots of its characteristic polynomial). The SVD is the usual representation of choice as it is stable, whereas the Jordan normal form is notoriously unstable to compute (see, e.g., [22, Chapter 7], [15, Chapter 4] and [19]). However, the SVD representation is not convenient when considering BPL algorithms, as $\mathcal{A}$ does not share the same singular vectors with powers of $\mathcal{A}$. Thus, in this paper, we choose to analyze our algorithm using the Jordan normal form. Admittedly, one should expect severe stability problems using such an approach. Surprisingly, we show that under mild conditions we manage to overcome these stability problems.

As before, we would like to approximate the generalized inverse $\mathcal{L}^\star$. There are two main issues to consider:

1. Peeling-off the 1-subspace. To do so, we need a good approximation of the stationary distribution of the corresponding random walk. In the undirected case, it can be easily inferred (i.e., in L) from the input. Here, we require it as an input to our algorithm.
2. Analyzing the convergence of the Taylor series of $(\mathcal{I} - \mathcal{A})^{-1}$ for a non diagonalizable $\mathcal{A}$. Recall that when a function f acts on a diagonalizable matrix $\mathcal{A}$, it acts on its eigenvalues in the natural way. In the non diagonalizable case, f acts on a Jordan *block*, which might have a large dimension, and although an eigenvalue λ on the diagonal is still mapped to an eigenvalue $f(\lambda)$, the structure of the rest of the block is no longer maintained, so we need to give this issue further consideration.

To address the second issue above, we use the theory of matrix functions that tells us exactly what $f(\mathbf{A})$ is. It turns out that there is a direct connection between $f(\mathbf{A})$, the dimension of the Jordan block, and the derivatives of f on the corresponding eigenvalue. Exploiting this connection, we manage to bound the number of terms in the Taylor series that is sufficient for convergence. The caveat here is that two "stability" parameters enter the picture. First, the spectral gap (whose formal definition we defer), which for directed graphs may no longer be at most polynomially-small and naturally affect the performance of our algorithm. Second, we also need the Jordan basis matrix V of $\mathcal{L}$ to be well-conditioned. We prove:

▶ **Theorem 2** (Informal). *There exists a probabilistic algorithm that gets as input an $n \times n$ stochastic matrix $\mathcal{S}$, desired accuracy and confidence parameters $\varepsilon, \delta > 0$, $\gamma > 0$ which is a lower-bound on the spectral gap of $\mathcal{S}$, κ which is an upper bound on the condition number of the Jordan basis of $\mathcal{S}$, and outputs with probability at least $1 - \delta$ an approximation of $\mathcal{L}^\star = (\mathcal{I} - \mathcal{S})^\star$ to within an ε-accuracy, using*

$$O\left(\log \frac{n}{\gamma\varepsilon} + \log\log \frac{\kappa}{\delta}\right)$$

space.

Remarkably, the dependency of the space complexity on the condition number of the Jordan basis matrix is *doubly-logarithmic*. This also allows us to show our algorithm operates well on operators for which the eigenvalues are polynomially far apart (see Theorem 29).

Having this theorem we show that in addition to undirected graphs, our approximation algorithm works for well-conditioned regular and Eulerian directed graph (which we know have a non-negligible spectral gap and their stationary distribution is fully-explicit) and general well-conditioned rapidly-mixing directed graphs. We thus see that the algorithm manages to approximate the solution of Laplacian systems over a large (and natural) class of directed graphs.

We conclude with a more philosophical note. In recent years we have seen several results showing that some natural linear-algebraic tasks capture the strength of various space-bounded models of computation. Results along this line are:

1. Ta-Shma [37] showed that it is possible to approximate the SVD of any matrix, and in particular to approximate its inverse, in BQL, with *polynomially-small* accuracy.[4] As no classical analogue is known, this result is one of the very few cases where a natural problem is known to lie in BQL but is not known to be in BPL.
2. Doron et al. [16] gave a BPL algorithm that computes the eigenvalues of *stochastic* matrices having real eigenvalues with *constant* accuracy. Moreover, they gave a linear-algebraic problem which is *complete* for BPL – roughly speaking, approximating, to polynomially-small accuracy, the second eigenvalue of a stochastic matrix (whose eigenvalues are not necessarily real).
3. Fefferman and Lin [18] gave two complete problems for BQL – approximating the inverse and the minimum eigenvalue of positive semi-definite matrices (both to polynomially-small accuracy).

We hence see that the deterministic, probabilistic and quantum space-bounded complexity classes can be roughly characterized by linear-algebraic promise problems, where the difference between the classes lies in the family of operators they can handle, being Hermitian, stochastic or general operators. The *exact* computation can be done in $\mathsf{DET} \subseteq \mathsf{NC}^2 \subseteq \mathsf{DSPACE}(O(\log^2 n))$. Our result is in line with the above, showing that approximating with polynomially-small accuracy the generalized inverse of a large class of *stochastic* matrices is in BPL.

[4] Roughly, BQL stands for the class of languages for which there exists an L-uniform family of quantum circuits solving it with only $O(\log n)$ qubits. It is known that $\mathsf{BQL} \subseteq \mathsf{NC}^2$ [40].

2 Preliminaries

2.1 Basic facts from linear algebra

For a matrix $\mathcal{A} \in \mathbb{C}^{n \times n}$, $\mathcal{A}^\dagger$ is its conjugate transpose. When it might not be clear from the context, for a vector $v \in \mathbb{C}^n$, we denote $|v\rangle$ as the column vector and $\langle v|$ as the row vector, so $\langle u | v \rangle$ is a scalar and $|v\rangle\langle u|$ is a rank-one matrix.

Every matrix $\mathcal{A}$ has a *singular value decomposition* (SVD) $\mathcal{A} = U \mathbf{\Sigma} V^\dagger$, where U and V are unitary and $\mathbf{\Sigma}$ is a diagonal matrix with non-negative entries, known as the singular values of $\mathcal{A}$.

The *spectrum* of a matrix $\mathcal{A}$, denoted $\mathsf{Spec}(\mathcal{A})$, is its set of (complex or real) eigenvalues. The *spectral radius* $\rho(\mathcal{A})$ of $\mathcal{A}$ is the largest absolute value of its eigenvalues. The *operator norm* $\|\mathcal{A}\|$ is $\max_{\|x\|_2=1} \|\mathcal{A}x\|$, which is also the largest singular value of $\mathcal{A}$. Notice that it is possible for $\|\mathcal{A}\|$ to be strictly larger than $\rho(\mathcal{A})$. The operator norm is sub-multiplicative. When $\mathcal{A}$ is invertible, $\kappa(\mathcal{A}) = \|\mathcal{A}\| \left\|\mathcal{A}^{-1}\right\|$ is its *condition number.* Also, we denote $\|\mathcal{A}\|_\infty$ as the induced ℓ_∞ norm, that is $\|\mathcal{A}\|_\infty = \max_{i \in [n]} \sum_{j \in [n]} |\mathcal{A}[i,j]|$. It holds that $\|\mathcal{A}\|_\infty \leq \sqrt{n}\, \|\mathcal{A}\|$.

For an eigenvalue λ of $\mathcal{A}$, a λ-right-eigenvector (or simply an eigenvector with eigenvalue λ) is a vector v such that $\mathcal{A}v = \lambda v$. A λ-left-eigenvector is a vector v such that $v^\dagger \mathcal{A} = \lambda v^\dagger$. We define the spectral gap $\gamma(\mathcal{A}) = 1 - \max_{\lambda \in \mathsf{Spec}(\mathcal{A}), \lambda \neq 1} |\lambda|$. Note that $\gamma(\mathcal{A}) \leq \min_{\lambda \in \mathsf{Spec}(\mathcal{A}), \lambda \neq 1} |1 - \lambda|$.

We denote by $\mathbf{1}$ the column vector of all ones and similarly $\mathbf{0}$ the column vector of all zeros.

2.2 The Perron-Frobenius theorem

The *underlying graph* of a matrix $\mathcal{A}$ has an edge (i,j) iff $\mathcal{A}[i,j] \neq 0$. A matrix $\mathcal{A}$ is *irreducible* if its underlying directed graph is strongly connected. When $\mathcal{A}$ is irreducible, its *period* is the greatest common divisor of the lengths of the closed directed paths in the underlying directed graph of $\mathcal{A}$. We say that $\mathcal{A}$ is *aperiodic* if its period is 1. A matrix $\mathcal{A}$ is *non-negative* if all its entries are non-negative, and it is *stochastic* if it is non-negative and every row sums to 1. We will need the Perron-Frobenius theorem for irreducible non-negative matrices (see, e.g., [21, Chapter 8]).

▶ **Theorem 3.** *Let $\mathcal{A}$ be an irreducible non-negative $n \times n$ matrix with period h and spectral radius $\rho(A) = r$. Then:*

1. *There exists an r-right-eigenvector v_1 and an r-left-eigenvector u_1 whose components are all positive.*
2. *$\mathcal{A}$ has exactly h complex eigenvalues with absolute value r and each one of them is a product of r with a different h-th root of unity. Consequently, if $\mathcal{A}$ is aperiodic then r is a simple eigenvalue, and all other eigenvalues have absolute value strictly smaller than r.*
3. *It holds that $\lim_{k \to \infty} \mathcal{A}^k / r^k = |v_1\rangle\langle u_1|$, where v_1 and u_1 are normalized so that $\langle v_1 | u_1 \rangle = 1$.*

If $\mathcal{A}$ is stochastic then $r = 1$. Furthermore, if $\mathcal{A}$ is stochastic, irreducible and aperiodic then v_1 is the all-ones vector $\mathbf{1}$ and $u_1 = \pi$ is the stationary distribution *of the corresponding random walk (all up to normalizations).*

2.3 Jordan normal form

▶ **Fact 4.** *Every complex $n \times n$ matrix $\mathcal{A}$ can be expressed in a Jordan normal form $\mathcal{A} = V\mathbf{A}V^{-1}$ where $\mathbf{A} = \mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_B)$,*

$$\mathbf{A}_b = \mathbf{A}_b(\lambda_b) = \begin{pmatrix} \lambda_b & 1 & & \\ & \lambda_b & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_b \end{pmatrix} \in \mathbb{C}^{\dim_b \times \dim_b},$$

and $\dim_1 + \ldots + \dim_b = n$. *The Jordan matrix $\mathbf{A}$ has the eigenvalues of $\mathcal{A}$ on its diagonal, and is unique up to the ordering of the blocks $\mathbf{A}_b$. For an eigenvalue λ_b, its algebraic multiplicity is the number of times it appears on the diagonal $\mathbf{A}$ and its geometric multiplicity is the number of blocks having λ_b on their diagonal. We say an eigenvalue is* simple *if its algebraic multiplicity is one.*

▶ **Claim 5** ([9], Chapter 3)**.** *Let $\mathcal{A}$ be an $n \times n$ complex matrix and let $\mathcal{A} = V\mathbf{A}V^{-1}$ be the Jordan normal form of $\mathcal{A}$, where $\mathbf{A} = \mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_B)$. Then, every Jordan block $\mathbf{A}_b$ corresponds to an $\mathcal{A}$-invariant subspace $E_b = \mathsf{Ker}\left((\lambda_b \mathcal{I} - \mathcal{A})^{\dim_b}\right)$ of dimension* $\dim_b$. *This gives a decomposition $\mathbb{C}^n = \bigoplus_{b=1}^{B} E_b$.*

For a Jordan decomposition $\mathcal{A} = V\mathbf{A}V^{-1}$, we will often write $\mathcal{A} = \sum_{b=1}^{B} V_b \mathbf{A}_b U_b$, where $\mathbf{A}_b$ is the b-th Jordan block, V_b are the columns of V that correspond to this block and similarly U_b are the rows of V^{-1} that correspond to this block.

When the operator is irreducible, aperiodic and stochastic, we can express the Perron-Frobenius theorem in the Jordan terminology and get:

▶ **Claim 6.** *Let $\mathcal{S}$ be an irreducible, aperiodic and stochastic matrix with a stationary distribution π so that $\langle \mathbf{1} | \pi \rangle = 1$ and let $\mathcal{S} = \sum_{b=1}^{B} V_b \mathbf{S}_b U_b$ be a Jordan decomposition of $\mathcal{S}$. Then,*

- $\mathbf{S}_1 = (1)$, *the 1×1 matrix with an entry 1.*
- *For all $b \geq 2$, $U_b V_1 = U_b \left|\mathbf{1}\right\rangle = \mathbf{0}$ and $U_1 V_b = \langle \pi | V_b = \mathbf{0}^\dagger$. Also, $\sum_{b=1}^{B} V_b U_b = \mathcal{I}$.*
- $V_1 \mathbf{S}_1 U_1 = |\mathbf{1}\rangle \langle \pi|$ *so* $\mathcal{S} = |\mathbf{1}\rangle \langle \pi| + \sum_{b=2}^{B} V_b \mathbf{S}_b U_b$.

Proof. If v is a (right) eigenvector of $\mathcal{S}$ with eigenvalue λ then $v \in \mathsf{Im}(\cup_{b:\lambda_b = \lambda} V_b)$. Similarly, if w is a left eigenvector of $\mathcal{S}$, then its eigenvalue is an eigenvalue of $\mathcal{S}$ and $w \in \mathsf{Im}(\cup_{b:\lambda_b = \lambda} U_b)$ (this is because $\mathcal{A}$ and $\mathcal{A}^\dagger$ have the same spectrum, see, e.g., [8, Chapter 9]).

Now, since $\mathcal{S}$ is stochastic, $\mathbf{1}$ is a 1-eigenvector. Also, there is a 1-left-eigenvector that we denote by π, and we normalize π such that $\langle \pi | \mathbf{1} \rangle = 1$. Furthermore, by the Perron-Frobenius theorem, the 1-eigenvalue is simple, so $\mathbf{S}_1 = (1)$, U_1 is a $1 \times n$ matrix and V_1 is a $n \times 1$ matrix. Furthermore, by the above, $\pi \in \mathsf{Im}(U_1)$, and since the dimension of the image is 1, we must have $\mathsf{Im}(U_1) = \mathrm{Span}(\{\pi\})$. Similarly, $\mathsf{Im}(V_1) = \mathrm{Span}(\{\mathbf{1}\})$. This completes the proof of the first item.

For the second item, let $U = V^{-1}$ and observe that since $UV = \mathcal{I}$, $\langle u_i | v_j \rangle = \delta_{i,j}$ (where u_i is the i-th row of U and v_j is the j-th column of V). Now, consider $b \neq b'$ and the product $P = U_b V_{b'}$. Every entry of P is of the form $\langle u_{b,i} | v_{b',j} \rangle$ where $i \in [\dim_b]$ and $j \in [\dim_{b'}]$. By the previous observation, they are all zeros. Also, $\mathcal{I}$ has a Jordan decomposition $V\mathbf{I}U$, so immediately it is clear that $\sum_{b=1}^{B} V_b U_b = \mathcal{I}$.

For the third item, Suppose $V_1 = \alpha\mathbf{1}$ and $U_1 = \beta\langle\pi|$ for some nonzero $\alpha, \beta \in \mathbb{C}$. We see that $V_1\mathbf{S}_1U_1 = \alpha\beta\,|\mathbf{1}\rangle\langle\pi|$. We want to determine $\alpha\beta$. Since $\langle\pi|\,\mathcal{S} = \langle\pi|$ we have that

$$\begin{aligned}\langle\pi| = \langle\pi|\,\mathcal{S} = \beta^{-1}U_1\mathcal{S} = \beta^{-1}U_1\sum_{b=1}^{B} V_b\mathbf{S}_bU_b \\ = \beta^{-1}U_1V_1\mathbf{I}_1U_1 + \beta^{-1}\sum_{b=2}^{B} U_1V_b\mathbf{S}_bU_b = \beta^{-1}\beta\alpha\beta\,\langle\pi|\,\mathbf{1}\rangle\,\langle\pi| \;=\; \alpha\beta\,\langle\pi|\,,\end{aligned}$$

so $\alpha\beta = 1$. Hence, $V_1\mathbf{S}_1U_1 = V_1U_1 = |\mathbf{1}\rangle\langle\pi|$. ◀

2.4 Functions of matrices

This subsection follows the book of Higham [23]. In the Jordan basis, each Jordan block is a matrix with some complex value λ over the main diagonal and 1 in the diagonal above it. We want to distinguish upper triangular matrices in which elements on the same diagonal have the same value. We note that this class D of matrices is closed under matrix addition and multiplication. We denote:

▶ **Definition 7.** For $0 \le i \le n-1$ let $\mathcal{D}_{n,i}$ be the $n \times n$ matrix that has 1 over the i-th diagonal and 0 elsewhere, where the 0-th diagonal is the main diagonal and the i-th diagonal is the diagonal i elements above it.

Clearly $D = \text{Span}\{\mathcal{D}_{n,0}, \ldots, \mathcal{D}_{n,n-1}\}$ is closed under matrix addition. Also, since

$$\mathcal{D}_{n,i} \cdot \mathcal{D}_{n,j} \;=\; \mathcal{D}_{n,i+j},$$

D is also closed under matrix multiplication.

Suppose $p \in \mathbb{C}[x]$ is a polynomial $p(x) = \sum_{i=0}^{d} c_i x^i$. We can evaluate the polynomial over the ring $M_n(\mathbb{C})$, i.e., given an $n \times n$ matrix $\mathcal{A}$ we let

$$p(\mathcal{A}) = \sum_{i=0}^{d} c_i\mathcal{A}^i.$$

Note that if $\mathcal{A} = V\mathbf{A}V^{-1}$ then $p(\mathcal{A}) = Vp(\mathbf{A})V^{-1}$. Also, if $\mathbf{A} = \mathsf{diag}(\mathbf{A}_1, \ldots, \mathbf{A}_B)$ then $p(\mathbf{A}) = \mathsf{diag}(p(\mathbf{A}_1), \ldots, p(\mathbf{A}_B))$. In the extreme case where $\mathcal{A}$ is diagonalizable and all Jordan blocks have dimension 1, we see that p acts on the eigenvalues of $\mathcal{A}$. In the general case, we need to understand how p acts on a Jordan block $\mathbf{A}_b = \lambda_b\mathbf{I} + \mathcal{D}_{\dim_b,1}$. The answer is quite surprising and holds for arbitrary differentiable functions.

▶ **Lemma 8** ([23], Chapter 1). *Let $f : \mathbb{C} \to \mathbb{C}$ and suppose it is differentiable n times on* $\mathsf{Spec}(\mathcal{A})$. *Let $\mathbf{A} \in \mathbb{C}^{n\times n}$ be a Jordan block $\mathbf{A} = \lambda\mathbf{I} + \mathcal{D}_{n,1}$. Then,*

$$f(\mathbf{A}) \;=\; \begin{pmatrix} f(\lambda) & f'(\lambda) & \ldots & \frac{f^{(n-1)}(\lambda)}{(n-1)!} \\ & f(\lambda) & \ddots & \vdots \\ & & \ddots & f'(\lambda) \\ & & & f(\lambda) \end{pmatrix} \;=\; \sum_{t=0}^{n-1} \frac{f^{(t)}(\lambda)}{t!}\mathcal{D}_{n,t}.$$

2.5 The generalized inverse

Let $\mathcal{A}$ be any complex linear operator. A generalized (reflexive) inverse $\mathcal{A}^+$ of $\mathcal{A}$ is a matrix that satisfies both $\mathcal{A}\mathcal{A}^+\mathcal{A} = \mathcal{A}$ and $\mathcal{A}^+\mathcal{A}\mathcal{A}^+ = \mathcal{A}^+$. A generalized inverse is not unique, however if we further demand that both $\mathcal{A}\mathcal{A}^+$ and $\mathcal{A}^+\mathcal{A}$ are Hermitian, then such an operator is unique, and is called the Moore-Penrose pseudo-inverse and can be computed using the singular values decomposition (SVD). If $\mathcal{A} = U\mathbf{\Sigma}V^\dagger$ is the SVD of $\mathcal{A}$ then the pseudo-inverse is $\mathcal{A}^+ = V\mathbf{\Sigma}^+U^\dagger$ where $\mathbf{\Sigma}^+ = \mathsf{inv}(\mathbf{\Sigma})$ and $\mathsf{inv}(x)$ is the univariate function that is $1/x$ when $x \neq 0$ and 0 otherwise.

We will not work with the SVD but rather with the Jordan canonical form. Let $\mathcal{A} = V\mathbf{A}V^{-1}$ be a Jordan decomposition of a singular matrix $\mathcal{A}$. When the algebraic multiplicity of the eigenvalue 0 is one, the matrix $\mathcal{A}^\star = \mathsf{inv}(A)$, according to Subsection 2.4, is well defined. Namely, $\mathsf{inv}(A) = V\mathbf{A}^{\mathsf{inv}}V^{-1}$ where $\mathbf{A}^{\mathsf{inv}}$ is obtained by inverting every Jordan block that does not correspond to the zero eigenvalue. It is immediate that $\mathcal{A}^\star$ is a generalized inverse, although it does not generally coincide with the pseudo-inverse. From here onward, we denote $\mathcal{A}^\star$ as the generalized inverse $\mathsf{inv}(A)$.

Any generalized inverse $\mathcal{A}^\star$ can be used to determine if a system of linear equations has any solution (and if so, to give them all). More concretely, if the system $\mathcal{A}x = b$ has a solution then all its solution are given by $x = \mathcal{A}^\star b + (I - \mathcal{A}^\star\mathcal{A})w$ for an arbitrary w. All of the above claims can be found, e.g., in [4].

It will later be evident that when $\mathcal{A} = \mathcal{L} = \mathcal{I} - \mathcal{S}$ is a Laplacian corresponding to an irreducible, aperiodic and stochastic matrix $\mathcal{S}$ with a stationary distribution π, the expression $\mathcal{I} - \mathcal{A}^\star\mathcal{A}$ is simply $|\mathbf{1}\rangle\langle\pi|$. Thus, if we find $\mathcal{L}^\star$ we can solve any set of equations $\mathcal{L}x = b$ that has a solution. In fact, this also works when we try to solve the system $\mathcal{L}x = b$ for b that does not admit any perfect solution, but is close to a vector in $\mathsf{Im}(\mathcal{L})$. To see that, say b is arbitrary, and on input b and $\mathcal{L}$ we output $z = \mathcal{L}^\star b$. Then $\|\mathcal{L}z - b\| = \|(\mathcal{L}\mathcal{L}^\star - \mathcal{I})b\| = \||\mathbf{1}\rangle\langle\pi|\, b\| = \sqrt{n} \cdot |\langle\pi, b\rangle|$, and so if b is δ close to being perpendicular to π (and so close to being in $\mathsf{Im}(\mathcal{L})$) then the solution $z = \mathcal{L}^\star b$ is such that $\mathcal{L}z$ is $\sqrt{n}\delta$ close to the desired value b.

2.6 Space-bounded probabilistic computation

2.6.1 The model of computation

A space-bounded probabilistic Turing machine has four semi-infinite tapes: a read-only *input tape*, a *work tape*, a read-only uni-directional *random-coins tape* and a write-only uni-directional *output tape*. We say a language is accepted by a probabilistic TM if for every input in the language the acceptance probability is at least 2/3 and for every input not in the language it is at most 1/3. As usual, the acceptance probability can be amplified as long as there is some non-negligible gap between the acceptance probability of yes and no instances.

The complexity class BPL comprises all languages accepted by a space-bounded probabilistic TM with space complexity $O(\log n)$ and polynomial time.

2.6.2 Simulatable matrices

We are often interested in approximating a *value* (e.g., a matrix entry) with probabilistic machines. Assume that for an input $x \in \{0,1\}^n$ there exists a value $u = u(x) \in \mathbb{C}$. We say a probabilistic TM (ε, δ)-*approximates* $u(x)$ if

$$\forall_{x \in \{0,1\}^n} \ \Pr_y[|M(x,y) - u(x)| \geq \varepsilon] \ \leq \ \delta.$$

If u is multi-valued (say, a vector) we say a TM (ε, δ)-approximates u if given an index i it (ε, δ)-approximates $u[i]$.

▶ **Definition 9.** We say that a family of matrices $\mathfrak{A}$ is *simulatable* if there exists a probabilistic algorithm that on input $\mathcal{A} \in \mathfrak{A}$ of dimension n, $k \in \mathbb{N}$, $s, t \in [n]$, $\varepsilon, \delta > 0$ runs in space $O(\log \frac{nk}{\varepsilon} + \log\log \frac{1}{\delta})$ and (ε, δ)-approximates $A^k[s,t]$.

Probabilistic logspace machines can approximate random walks well. In [17], it is shown that:

▶ **Lemma 10.** *The family of stochastic matrices is simulatable.*

We can also conclude:

▶ **Lemma 11** ([17]). *Let $A \in \mathbb{C}^{n\times n}$ be a stochastic matrix and let $p = \sum_{i=0}^{d} c_i x^i$ be a complex polynomial such that:*

- *For every i, $|c_i| \le M$, and,*
- *The coefficients c_i are explicit in the sense that there exists an algorithm that given $k \le d, \varepsilon, \delta$ outputs an (ε, δ)-approximation of c_k using $O(\log \frac{nMd \log \frac{1}{\delta}}{\varepsilon})$ space.*

Then, the entries of $p(A)$ can be (ε, δ)-approximated using $O(\log \frac{nMd \log \frac{1}{\delta}}{\varepsilon})$ space.

3 Approximating $(\mathcal{I} - \mathcal{A})^{-1}$ by the Taylor series

We start with the simple case of normal matrices, and consider general functions.

▶ **Theorem 12.** *Let $f, p : \mathbb{C} \to \mathbb{C}$ and $\varepsilon > 0$. Suppose $\mathcal{A}$ is a normal matrix such that for every $\lambda \in \mathsf{Spec}(A)$, $|f(\lambda) - p(\lambda)| \le \varepsilon$. Then, $\|f(\mathcal{A}) - p(\mathcal{A})\| \le \varepsilon$.*

Proof. $\mathcal{A}$ is normal, so it is diagonalizable by a unitary matrix, $\mathcal{A} = UDU^{\dagger}$. Also, $f(\mathcal{A}) = Uf(D)U^{\dagger}$ and $p(\mathcal{A}) = Up(D)U^{\dagger}$. Thus, we have that

$$\|f(\mathcal{A}) - p(\mathcal{A})\| \;\le\; \|U\|\,\|U^{\dagger}\|\,\|f(D) - p(D)\| \;=\; \|f(D) - p(D)\|\,,$$

and $\|f(D) - p(D)\|$ is simply $\max_{\lambda \in \mathsf{Spec}(\mathcal{A})} |f(\lambda) - p(\lambda)| \le \varepsilon$. ◀

With that we can easily see that when $\mathcal{A}$ is normal, $\sum_{i=0}^{T} \mathcal{A}^i$ approximates $(\mathcal{I} - \mathcal{A})^{-1}$ pretty well. Formally,

▶ **Corollary 13.** *Let $\mathcal{A}$ be a normal matrix and suppose $\mathsf{Spec}(\mathcal{A}) \subseteq [0,1)$ and in particular $\mathcal{I} - \mathcal{A}$ is invertible. Then,*

$$\left\| (\mathcal{I} - \mathcal{A})^{-1} - \sum_{i=0}^{T} \mathcal{A}^i \right\| \;\le\; \frac{e^{-T\overline{\lambda}(\mathcal{A})}}{\overline{\lambda}(\mathcal{A})}.$$

Proof. For $\lambda \in [0,1)$, it holds that

$$\left| \frac{1}{1-\lambda} - \sum_{i=0}^{T} \lambda^i \right| \;\le\; \sum_{T+1}^{\infty} \lambda^i \;=\; \frac{\lambda^{T+1}}{1-\lambda}.$$

The above expression is maximized where $\lambda = 1 - \gamma(\mathcal{A})$, so we have:

$$\left| \frac{1}{1-\lambda} - \sum_{i=0}^{T} \lambda^i \right| \;\le\; \frac{(1-\gamma(\mathcal{A}))^T}{\gamma(\mathcal{A})} \;\le\; \frac{e^{-T\gamma(\mathcal{A})}}{\gamma(\mathcal{A})},$$

and the corollary follows. ◀

We would like to extend this result to arbitrary operators $\mathcal{A}$. As a first attempt we begin with generalizing Theorem 12 to arbitrary operators. For that we need the representation of $\mathcal{A}$ in its Jordan normal form, and we also need the function p and its derivatives to approximate the target function f and its derivatives well. We prove:

▶ **Theorem 14.** *Let $f, p : \mathbb{C} \to \mathbb{C}$. Suppose $\mathcal{A}$ is an $n \times n$ matrix such that for every $\lambda \in \mathsf{Spec}(A)$ and every $k \le n$, $|f^{(k)}(\lambda) - p^{(k)}(\lambda)| \le k! \cdot \varepsilon_k$. Furthermore, assume $\mathcal{A}$ has a Jordan decomposition $\mathcal{A} = V\mathbf{A}V^{-1}$, and the largest Jordan block has dimension D. Then, $\|f(\mathcal{A}) - p(\mathcal{A})\| \le \kappa(V) \cdot \sum_{k=0}^{D-1} \varepsilon_k$.*

Proof. Let $\mathbf{A} = \mathbf{A}_1 \oplus \ldots \oplus \mathbf{A}_b$, corresponding to the different Jordan blocks. By Lemma 8, $f(\mathcal{A}) = Vf(\mathbf{A})V^{-1}$ where $f(\mathbf{A}) = f(\mathbf{A}_1) \oplus \ldots \oplus f(\mathbf{A}_b)$,

$$f(\mathbf{A}_i) \;=\; \begin{pmatrix} f(\lambda_i) & f'(\lambda_i) & \ldots & \frac{f^{(\dim_i - 1)}(\lambda_i)}{(\dim_i - 1)!} \\ & f(\lambda_i) & \ddots & \vdots \\ & & \ddots & f'(\lambda_i) \\ & & & f(\lambda_i) \end{pmatrix} \;=\; \sum_{k=0}^{\dim_i - 1} \frac{f^{(k)}(\lambda_i)}{k!} \mathcal{D}_{\dim_i, k},$$

and λ_i is the eigenvalue corresponding to the block $\mathbf{A}_i$ of dimension $\dim_i$. The same of course holds for p. Thus,

$$\|f(\mathcal{A}) - p(\mathcal{A})\| \;=\; \left\|V(f(\mathbf{A}) - p(\mathbf{A}))V^{-1}\right\| \;\le\; \kappa(V) \cdot \|f(\mathbf{A}) - p(\mathbf{A})\|\,.$$

To bound the latter expression, note that

$$\begin{aligned} \|f(\mathbf{A}) - p(\mathbf{A})\| &= \max_{i \in [b]} \|f(\mathbf{A}_i) - p(\mathbf{A}_i)\| \\ &\le \max_{i \in [b]} \sum_{k=0}^{\dim_i - 1} \left| \frac{f^{(k)}(\lambda_i) - p^{(k)}(\lambda_i)}{k!} \right| \|\mathcal{D}_{\dim_i, k}\| \;\le\; \sum_{k=0}^{D-1} \varepsilon_k. \end{aligned}$$ ◀

When $\mathcal{A}$ is normal, $\kappa(V) = 1$ and the maximal block length is 1, so we recover Theorem 12. We now check what we get for $(\mathcal{I} - \mathcal{A})^{-1}$ and an arbitrary operator $\mathcal{A}$:

▶ **Corollary 15.** *Suppose $\mathcal{A}$ is an $n \times n$ matrix that has a Jordan decomposition $\mathcal{A} = V\mathbf{A}V^{-1}$. Suppose every eigenvalue λ of $\mathcal{A}$ satisfies $|\lambda| < 1$ and in particular $\mathcal{I} - \mathcal{A}$ is invertible. Let $T \in \mathbb{N}$ such that $T \ge \frac{8n^2}{\gamma(\mathcal{A})^2}$, let $f(\mathcal{A}) = (\mathcal{I} - \mathcal{A})^{-1}$ and $p(\mathcal{A}) = \sum_{i=0}^{T} \mathcal{A}^i$. Then,*

$$\|f(\mathcal{A}) - p(\mathcal{A})\| \;\le\; 2n\kappa(V) \frac{e^{-T\gamma(\mathcal{A})/4}}{\gamma(\mathcal{A})}.$$

Proof. Let $\mathcal{A}$ be an $n \times n$ matrix and suppose every eigenvalue λ of $\mathcal{A}$ satisfies $|\lambda| < 1$. We consider, again, inverting $\mathcal{I} - \mathcal{A}$ by considering the function $f(\lambda) = \frac{1}{1-\lambda}$ and its power-series expansion $p(\lambda) = \sum_{i=0}^{T} \lambda^i$. For $k \le n$, one can verify that $\frac{1}{k!} f^{(k)}(\lambda) = \frac{1}{(1-\lambda)^{k+1}}$ and $\frac{1}{k!} p^{(k)}(\lambda) = \sum_{i=0}^{T-k} \binom{k+i}{k} \lambda^i$. Also, $\frac{1}{k!} f^{(k)}(\lambda) = \sum_{i=0}^{\infty} \binom{k+i}{k} \lambda^i$ so we see that

$$\varepsilon_k \;=\; \left| \sum_{i=T-k+1}^{\infty} \binom{k+i}{k} \lambda^i \right|.$$

As $T \ge 4n$, $T - k + 1 \ge T/2$. Also, $\binom{k+i}{k} \le (k+i)^k \le (2i)^k$, so $\varepsilon_k \;\le\; \sum_{i=T/2}^{\infty} (2i)^k \lambda^i$. Now, we have that $(2i)^k \leq |\lambda|^{-i/2}$, since

$$\begin{aligned}(2i)^k|\lambda|^{i/2} &= e^{k\ln(2i)-(i/2)\ln\frac{1}{|\lambda|}} = e^{\frac{1}{2}\left(2k\ln(2i)-i\ln\frac{1}{|\lambda|}\right)} \leq e^{\frac{1}{2}\left(n\sqrt{i}-i\ln\frac{1}{|\lambda|}\right)}\\ &\leq e^{\frac{\sqrt{i}}{2}\left(n-\sqrt{i}\ln\frac{1}{|\lambda|}\right)} \leq e^{\frac{\sqrt{i}}{2}\left(n-\sqrt{T/2}\cdot\ln\frac{1}{1-\gamma(\mathcal{A})}\right)} \leq e^{\frac{\sqrt{i}}{2}\left(n-\sqrt{T/2}\cdot\gamma(\mathcal{A})\right)}\\ &\leq e^{\frac{\sqrt{i}}{2}(n-2n)} \leq 1.\end{aligned}$$

Plugging it to the above bound for ε_k, we obtain:

$$\varepsilon_k \leq \left|\sum_{i=T/2}^{\infty}\lambda^{i/2}\right| = \left|\frac{\lambda^{T/4}}{1-\sqrt{\lambda}}\right|.$$

To bound $\left|\frac{1}{1-\sqrt{\lambda}}\right|$, we use the fact that:

$$\left|\frac{1}{1-\sqrt{\lambda}}\right| = \frac{|1+\sqrt{\lambda}|}{|1-\lambda|} \leq \frac{2}{\gamma(\mathcal{A})}.$$

Altogether,

$$\varepsilon_k \leq \frac{2}{\gamma(\mathcal{A})}(1-\gamma(\mathcal{A}))^{T/4} \leq \frac{2e^{-T\gamma(\mathcal{A})/4}}{\gamma(\mathcal{A})}.$$

The Corollary follows by applying Theorem 14 and using the fact that $D \leq n$. ◀

4 Computing the generalized inverse of the Laplacian

In this section we approximate the generalized inverse of the Laplacian of directed graphs as long as we have a good approximation of its stationary distribution. Formally,

▶ **Theorem 16.** *There exists a probabilistic algorithm that gets as input:*

- *An $n \times n$ irreducible, aperiodic stochastic matrix $\mathcal{S}$,*
- *Two parameters, κ and γ, which describe how stable the input $\mathcal{S}$ is:*
  - *Suppose $\kappa \geq \kappa(V)$, where $\mathcal{S} = V\mathbf{S}V^{-1}$ is any Jordan decomposition of $\mathcal{S}$, and,*
  - *$\gamma(\mathcal{S}) \geq \gamma$.*
- *Desired accuracy and confidence parameters $\varepsilon, \delta > 0$.*
- *An approximation $\tilde{\pi}$ of the stationary distribution π of $\mathcal{S}$, where $\|\tilde{\pi}-\pi\| \leq \tau$ and $\tau \leq \frac{\varepsilon}{(T+1)\sqrt{n}}$ for $T = \frac{8n^2}{\gamma^2}\left(1+\log\frac{n\kappa}{\varepsilon\gamma}\right)$.*

Let $\mathcal{L}$ denote the Laplacian, $\mathcal{L} = \mathcal{I} - \mathcal{S}$. Then, the algorithm outputs a $(3\varepsilon, \delta)$-approximation of $\mathcal{L}^\star$ using

$$O\left(\log\frac{n}{\gamma\varepsilon} + \log\log\frac{\kappa}{\delta}\right)$$

space.

Intuitively, we would like to employ the following approach. Given a stochastic operator $\mathcal{S}$ with a unique stationary distribution π, we would like to "peel off" the 1×1 Jordan block with eigenvalue 1, so that we are left with an operator $\mathcal{A}$ such that $\mathcal{I} - \mathcal{A}$ is invertible. Then, we would like to use Corollary 15 to approximate $(\mathcal{I} - \mathcal{A})^{-1}$ by $\sum_{i=0}^{T} \mathcal{A}^i$, using the fact that we can approximate $\mathcal{A}^i$ well with a BPL algorithm.

There are two obstacles that we need to overcome:

- First, when $\mathcal{S}$ in not normal, we do not have an orthonormal basis, so we need to explain what "peeling off" the stationary distribution means. It turns out that $\mathcal{A} = \mathcal{S} - |\mathbf{1}\rangle\langle\pi|$.
- Second, while $\mathcal{S}$ is stochastic, $\mathcal{A} = \mathcal{S} - |\mathbf{1}\rangle\langle\pi|$ is not, and furthermore, its ℓ_∞ norm is usually greater than 1. In particular, we cannot immediately assume that we can approximate high powers of it in BPL. We will show that $\mathcal{A}$ is still simulatable because $|\mathbf{1}\rangle\langle\pi|$ commutes with both $\mathcal{S}$ and $\mathcal{A}$.

We also need to check that the fact that $\tilde{\pi}$ is only close to π and not exactly it, does not affect the parameters by too much.

We start the formal exposition with a precise description of the algorithm.

4.1 The Algorithm

The algorithm first computes the parameter

$$T \;=\; \left\lceil \frac{8n^2}{\gamma^2}\left(1+\log\frac{n\kappa}{\varepsilon\gamma}\right)\right\rceil .$$

The algorithm then computes an (ε,δ)-approximation of the matrix

$$\widetilde{Q}_T(\mathcal{S}) \;=\; \left(\sum_{i=0}^{T}\mathcal{S}^i\right) - (T+1)\,|\mathbf{1}\rangle\langle\tilde{\pi}|$$

using Lemma 11 (note that since $\tilde{\pi}$ is given, we *approximate* the power series and compute $(T+1)\,|\mathbf{1}\rangle\langle\tilde{\pi}|$ exactly).

We first argue that the algorithm runs in small space and then analyze correctness.

4.2 Efficiency

We observe:

▶ **Lemma 17.** *For every $\varepsilon,\delta>0$ and integer T, and any $n\times n$ stochastic matrix $\mathcal{S}$, the entries of $\widetilde{Q}_T(\mathcal{S})$ can be (ε,δ)-approximated using $O\left(\log\frac{nT\log\frac{1}{\delta}}{\varepsilon}\right)$ space.*

Proof. The claim follows directly from Lemma 11 since $\mathcal{S}$ is stochastic. ◀

4.3 Correctness

We first do the analysis in the ideal situation that $\tilde{\pi}=\pi$ and see that in this case the algorithm $(2\varepsilon,\delta)$-approximates $\mathcal{L}^\star$. We then show that when $\|\pi-\tilde{\pi}\|\le\tau$ the algorithm $(3\varepsilon,\delta)$-approximates $\mathcal{L}^\star$.

4.3.1 Peeling off the 1-eigenspace

Throughout the proof we use the representation of $\mathcal{S}$ guaranteed by Claim 6. Namely, $\mathcal{S}$ can be written as $\mathcal{S}=\sum_{b=1}^{B}V_b\mathbf{S}_bU_b$ where

- $\mathbf{S}_1$ is a 1×1 matrix and $\mathbf{S}_1=(1)$. Also, $V_1U_1=|\mathbf{1}\rangle\langle\pi|$ and $\langle\mathbf{1}|\,\pi\rangle=1$,
- For all $b\ge2$, $U_b\,|\mathbf{1}\rangle=\mathbf{0}$ and $\langle\pi|\,V_b=\mathbf{0}^\dagger$, and
- $\sum_{b=1}^{B}V_bU_b=\mathcal{I}$.

Our goal is to find the generalized inverse of $\mathcal{L} = \mathcal{I} - \mathcal{S}$. As explained before, our first step is to "peel-off" from $\mathcal{S}$ the 1-eigenspace, and the correct way to do that is by annihilating the 1×1 Jordan block with eigenvalue 1. We therefore define:

$$\mathcal{A} \;=\; \mathcal{S} - |\mathbf{1}\rangle\langle\pi|\,.$$

We notice that $\mathcal{S}$, $\mathcal{A}$, $\mathcal{L}$ and $\mathcal{L}^\star$ share the same Jordan basis, therefore, if we express $\mathcal{S} = \sum_{b=1}^{B} U_b \mathbf{S}_b V_b$ then

$$\mathcal{L} \;=\; \sum_{b=2}^{B} V_b(\mathbf{I}_b - \mathbf{S}_b)U_b,$$

and,

$$\mathcal{A} \;=\; \sum_{b=2}^{B} V_b \mathbf{S}_b U_b.$$

We denote $\mathbf{L}_b = \mathbf{I}_b - \mathbf{S}_b$ for $b \geq 2$ (and $\mathbf{L}_1$ is the zero matrix). The big advantage of $\mathcal{A}$ over $\mathcal{S}$ is that in $\mathcal{A}$ all eigenvalues have magnitude smaller than 1, as $\mathcal{A} = \sum_{b=2}^{B} V_b \mathbf{S}_b U_b$, and therefore $\mathcal{I} - \mathcal{A}$ is invertible. We still need, however, to relate $\mathcal{L}^\star$ to $(\mathcal{I} - \mathcal{A})^{-1}$. We prove:

▶ **Lemma 18.** $\mathcal{L}^\star = (\mathcal{I} - \mathcal{A})^{-1} - |\mathbf{1}\rangle\langle\pi|$.

Proof. Recall that $\mathcal{S} = |\mathbf{1}\rangle\langle\pi| + \sum_{b=2}^{B} V_b \mathbf{S}_b U_b$, $\mathcal{A} = \sum_{b=2}^{B} V_b \mathbf{S}_b U_b$ and $\mathcal{I} = \sum_{b=1}^{B} V_b U_b$. Hence,

$$\mathcal{I} - \mathcal{A} \;=\; \sum_{b=1}^{B} V_b U_b - \sum_{b=2}^{B} V_b \mathbf{S}_b U_b \;=\; V_1 U_1 + \sum_{b=2}^{B} V_b(\mathbf{I}_b - \mathbf{S}_b)U_b \;=\; |\mathbf{1}\rangle\langle\pi| + \sum_{b=2}^{B} V_b \mathbf{L}_b U_b.$$

The inverse is thus given by

$$(\mathcal{I} - \mathcal{A})^{-1} \;=\; |\mathbf{1}\rangle\langle\pi| + \sum_{b=2}^{B} V_b \mathbf{L}_b^{-1} U_b \;=\; |\mathbf{1}\rangle\langle\pi| + \mathcal{L}^\star,$$

as desired. ◀

Intuitively, this means that approximating $(\mathcal{I} - \mathcal{A})^{-1}$ suffices for approximating $\mathcal{L}^\star$, and we next consider approximating $(\mathcal{I} - \mathcal{A})^{-1}$.

4.3.2 Approximating $(\mathcal{I} - \mathcal{A})^{-1}$

Since all eigenvalues of $\mathcal{A}$ have magnitude smaller than 1, we can apply Corollary 15 and get:

▶ **Lemma 19.**

$$\left\| (\mathcal{I} - \mathcal{A})^{-1} - \sum_{k=0}^{T} \mathcal{A}^k \right\| \;\leq\; \varepsilon.$$

Proof. We saw that $\mathcal{A} = \sum_{b=2}^{B} V_b \mathbf{S}_b U_b$, and by the Perron-Frobenius theorem the eigenvalues that are written on $\mathbf{S}_b$ for $b \geq 2$, are at most $1 - \gamma < 1$ in absolute value. Thus, all eigenvalues of $\mathcal{A}$ have absolute value at most $\gamma(\mathcal{S})$. By Corollary 15, for $T \geq \frac{8n^2}{\gamma(\mathcal{S})^2}$,

$$\left\| (\mathcal{I} - \mathcal{A})^{-1} - \sum_{k=0}^{T} \mathcal{A}^k \right\| \;\leq\; 2n\kappa(V)\frac{e^{-T\gamma(\mathcal{S})/4}}{\gamma(\mathcal{S})}.$$

Substituting $T = \left\lceil \frac{8n^2}{\gamma(\mathcal{S})^2} \ln \frac{2n\kappa(V)}{\varepsilon\gamma(\mathcal{S})} \right\rceil$, the desired bound holds. ◀

Thus, the problem now reduces to simulating $\mathcal{A}^i$ in small space. As mentioned before, $\mathcal{A}$ is not stochastic and its ℓ_∞ norm is often larger than 1. However $\mathcal{A} = \mathcal{S} - |\mathbf{1}\rangle\langle\pi|$ has a very special form that conforms with the Jordan basis structure, which we now employ:

▶ **Claim 20.** *The matrices $\mathcal{S}$ and $|\mathbf{1}\rangle\langle\pi|$ commute, and furthermore $\mathcal{S} \cdot |\mathbf{1}\rangle\langle\pi| = |\mathbf{1}\rangle\langle\pi| \cdot \mathcal{S} = |\mathbf{1}\rangle\langle\pi|$.*

Proof.

$$\mathcal{S} \cdot |\mathbf{1}\rangle\langle\pi| \ = \ |\mathbf{1}\rangle\langle\pi| + \sum_{b=2}^{B} V_b \mathbf{S}_b U_b \cdot |\mathbf{1}\rangle\langle\pi| \ = \ |\mathbf{1}\rangle\langle\pi|,$$

and,

$$|\mathbf{1}\rangle\langle\pi| \cdot \mathcal{S} \ = \ |\mathbf{1}\rangle\langle\pi| + \sum_{b=2}^{B} |\mathbf{1}\rangle\langle\pi| \cdot V_b \mathbf{S}_b U_b \ = \ |\mathbf{1}\rangle\langle\pi|. \qquad ◀$$

▶ **Claim 21.** *For every $k \geq 1$, $\mathcal{A}^k = \mathcal{S}^k - |\mathbf{1}\rangle\langle\pi|$.*

Proof. The proof is by induction on k. For $k = 1$ the claim follows by the definition. Assume the statement holds for $k \in \mathbb{N}$, and consider $\mathcal{A}^{k+1}$, so By Claim 20:

$$\begin{aligned}\mathcal{A}^{k+1} &= \ (\mathcal{S} - |\mathbf{1}\rangle\langle\pi|) \cdot (\mathcal{S}^k - |\mathbf{1}\rangle\langle\pi|) \\ &= \ \mathcal{S}^{k+1} - \mathcal{S} \cdot |\mathbf{1}\rangle\langle\pi| - |\mathbf{1}\rangle\langle\pi| \cdot \mathcal{S}^k + |\mathbf{1}\rangle\langle\pi|\mathbf{1}\rangle\langle\pi| \\ &= \ \mathcal{S}^{k+1} - |\mathbf{1}\rangle\langle\pi| - |\mathbf{1}\rangle\langle\pi| + |\mathbf{1}\rangle\langle\pi| \ = \ \mathcal{S}^{k+1} - |\mathbf{1}\rangle\langle\pi|. \qquad ◀\end{aligned}$$

Thus, $\mathcal{A}$ is simulatable and we can approximate $(\mathcal{I} - \mathcal{A})^{-1}$ in small space.

4.3.3 Putting everything together

Define the *ideal* polynomial Q_T by:

$$Q_T(\mathcal{S}) \ = \ \left(\sum_{i=0}^{T} \mathcal{S}^i\right) - (T+1)\,|\mathbf{1}\rangle\langle\pi|.$$

▶ **Lemma 22.** $\|\mathcal{L}^\star - Q_T(\mathcal{S})\| \leq \varepsilon$.

Proof.

$$\begin{aligned}\|\mathcal{L}^\star - Q_T(\mathcal{S})\| &= \ \left\|(\mathcal{I} - \mathcal{A})^{-1} - |\mathbf{1}\rangle\langle\pi| - Q_T(\mathcal{S})\right\| \\ &\leq \ \left\|\left(\sum_{i=0}^{T} \mathcal{A}^i\right) - |\mathbf{1}\rangle\langle\pi| - Q_T(\mathcal{S})\right\| + \varepsilon \\ &= \ \left\|\mathcal{A}^0 + \sum_{i=1}^{T} \left(\mathcal{S}^i - |\mathbf{1}\rangle\langle\pi|\right) - |\mathbf{1}\rangle\langle\pi| - Q_T(\mathcal{S})\right\| + \varepsilon \\ &= \ \left\|\left(\sum_{i=0}^{T} \mathcal{S}^i\right) - (T+1)\,|\mathbf{1}\rangle\langle\pi| - Q_T(\mathcal{S})\right\| + \varepsilon \ = \ \varepsilon. \qquad ◀\end{aligned}$$

Finally, we check how the fact that $\tilde{\pi}$ is only close to π, affects our accuracy. We see that:

▶ **Claim 23.** $\left\|\widetilde{Q}_T(\mathcal{S}) - Q_T(\mathcal{S})\right\| \leq \varepsilon$.

Proof. Notice that $\widetilde{Q}_T(\mathcal{S}) - Q_T(\mathcal{S}) = (T+1)\,|\mathbf{1}\rangle\langle\tilde{\pi} - \pi|$. Therefore, $\left\|\widetilde{Q}_T(\mathcal{S}) - Q_T(\mathcal{S})\right\| \leq (T+1) \cdot \|\mathbf{1}\| \cdot \|\tilde{\pi} - \pi\|$. The proof follows because $\|\mathbf{1}\| = \sqrt{n}$ and $\|\tilde{\pi} - \pi\| \leq \tau \leq \frac{\varepsilon}{\sqrt{n}(T+1)}$. ◀

Now, since we (ε, δ)-approximate $\widetilde{Q}_T(\mathcal{S})$, then except for probability δ what we output is ε-close to $\widetilde{Q}_T(\mathcal{S})$, and therefore it is 2ε-close to $Q_T(\mathcal{S})$ and 3ε-close to $\mathcal{L}^\star$, which completes the proof of Theorem 16.

5 Some specific families of graphs

Ultimately, we would like to solve in BPL any set of equations $\mathcal{L}x = b$, where b is close to $\mathsf{Im}(\mathcal{L})$, and where $\mathcal{L}$ is the Laplacian of a stochastic matrix $\mathcal{S}$. Theorem 16 is a step towards this goal, but it works only when:

- $\mathcal{S}$ is irreducible, namely, its underlying graph is strongly connected,
- $\mathcal{S}$ is aperiodic,
- We can approximate well the unique stationary distribution π,
- $\gamma(\mathcal{S}) \geq \frac{1}{n^a}$ for some constant a, i.e., all eigenvalues except the largest one, are at most $1-\gamma$ in absolute value, and,
- $\kappa(V) \leq 2^{n^b}$ for some constant b, where $\mathcal{S} = V\mathbf{S}V^{-1}$ is a Jordan decomposition and $\kappa(V) = \|V\| \cdot \left\|V^{-1}\right\|$. Notice that here we may tolerate exponential $\kappa(V)$ as the space complexity dependency on κ is doubly-logarithmic.

In this section we want to examine which requirements can be relaxed. The section is organized as follows. First, we note that we can get rid of the aperiodicity requirement and we can somewhat relax the spectral gap requirement. Then we show that in some cases we can get rid of the $\kappa(V)$ requirement (when the eigenvalues are polynomially separated). Finally, we give specific results for:

- Undirected graphs,
- Directed Eulerian graphs (which generalize directed regular graphs), and,
- Directed rapidly-mixing graphs.

5.1 Omitting the aperiodicity requirement using lazy walks

Given a stochastic matrix $\mathcal{S}$ we can convert it to the corresponding lazy walk $\mathcal{S}' = \frac{1}{2}(\mathcal{I} + \mathcal{S})$, that stays in place with probability half. Define:

$$\gamma'(\mathcal{S}) \;=\; \max_{\lambda \in \mathsf{Spec}(\mathcal{S}'), \lambda \neq 1} (1 - \Re(\lambda)).$$

The conversion has two benefits. First, the walk is clearly aperiodic. Also, we will be able to replace the condition $\gamma \leq \gamma(\mathcal{S})$, with the milder condition $\gamma \leq \gamma'(\mathcal{S})$. We will also show that we can recover the generalized inverse of the Laplacian of a graph G from that of the lazy walk variant of G. We prove:

▶ **Theorem 24.** *There exists a probabilistic algorithm that gets as input:*

- *An $n \times n$ irreducible, stochastic matrix $\mathcal{S}$.*
- *Two parameters, κ and γ, which describe how stable the input $\mathcal{S}$ is:*
  - *Suppose $\kappa \geq \kappa(V)$, where $\mathcal{S} = V\mathbf{S}V^{-1}$ is any Jordan decomposition of $\mathcal{S}$, and,*
  - *$\gamma'(\mathcal{S}) \geq \gamma$.*
- *Desired accuracy and confidence parameters $\varepsilon, \delta > 0$.*
- *An approximation $\tilde{\pi}$ of the stationary distribution π of $\mathcal{S}$, where $\|\tilde{\pi} - \pi\| \leq \tau$ and $\tau \leq \frac{\varepsilon}{(T+1)\sqrt{n}}$ for $T = \frac{8n^2}{\gamma^2}\left(1 + \log\frac{n\kappa}{\varepsilon\gamma}\right)$.*

Let $\mathcal{L}$ denote the Laplacian, $\mathcal{L} = \mathcal{I} - \mathcal{S}$. Then, the algorithm outputs a $(3\varepsilon, \delta)$-approximation of $\mathcal{L}^\star$ using

$$O\left(\log \frac{n}{\gamma\varepsilon} + \log\log \frac{\kappa}{\delta}\right)$$

space.

Proof. We run the algorithm of Theorem 16 over $\mathcal{S}' = \frac{1}{2}(\mathcal{I} + \mathcal{S})$. It is clear that $\mathcal{S}'$ is stochastic and aperiodic. By assumption, $\mathcal{S}'$ is irreducible (since $\mathcal{S}$ is). Also, $\mathcal{S}$ and $\mathcal{S}'$ have the same V and by assumption $\kappa(V) \le \kappa$. They also share the same stationary distribution π, and we are given π' which is close to π.

We will soon prove that $\gamma(\mathcal{S}') \ge \frac{\gamma'(\mathcal{S})}{4}$. Therefore, by Theorem 16, we get a $(3\varepsilon, \delta)$-approximation of $(\mathcal{I} - \mathcal{S}')^\star$. Finally, we will see that $(\mathcal{I} - \mathcal{S}')^\star = 2(\mathcal{I} - \mathcal{S})^\star$ and so we easily get an approximation for $(\mathcal{I} - \mathcal{S})^\star$.

To see that indeed $(\mathcal{I} - \mathcal{S}')^\star = 2(\mathcal{I} - \mathcal{S})^\star$, notice that $\mathcal{I}$ and $\mathcal{S}$ share the same Jordan basis V. The first block in $\mathcal{S}'$ and $\mathcal{S}$ is the same, and for $b \ge 2$, if the b-th block in $\mathcal{S}$ is $\mathbf{S}_b$, then the b-th block in $(\mathcal{I} - \mathcal{S}')^\star$ is $(\mathcal{I} - \frac{1}{2}(I + \mathbf{S}_b))^{-1} = 2(I - \mathbf{S}_b)^{-1}$ and the b-th block of $(\mathcal{I} - \mathcal{S})^\star$ is $(\mathbf{I} - \mathbf{S}_b)^{-1}$.

Thus, all that is left is to prove:

▶ **Claim 25.** *It holds that $\gamma(\mathcal{S}') \ge \frac{\gamma'(\mathcal{S})}{4}$.*

Proof. Fix $\lambda \in \mathsf{Spec}(\mathcal{S})$, $|\lambda| \le 1$, and write $\lambda = a + bi$ for $a, b \in \mathbb{R}$. Also, let $\lambda' = \frac{1}{2} + \frac{1}{2}\lambda = \frac{1+a}{2} + \frac{b}{2}i$, which is the corresponding eigenvalue in $\mathcal{S}'$. Thus:

$$|\lambda'|^2 \;=\; \frac{a^2 + b^2 + 2a + 1}{4} \;\le\; \frac{1 + 2a + 1}{4} \;=\; \frac{1 + \Re(\lambda)}{2},$$

so $1 - |\lambda'| \le 1 - \sqrt{\frac{1+\Re(\lambda)}{2}}$. The claim follows since for every R such that $|R| \le 1$, $1 - \sqrt{\frac{1+R}{2}} \ge \frac{1}{4}(1 - R)$. ◀

◀

5.2 Undirected graphs

Given an undirected graph we can easily partition it to its connected components using the fact that st-connectivity of undirected graphs is in BPL [1] (in fact, Reingold showed it is in L [30]). Therefore, we can solve the system of equations on each connected component separately.

Now, say we are given an undirected graph G and $\mathcal{A}$ is its adjacency matrix. The stochastic matrix $\mathcal{S}$ associated with G is $D^{-1}\mathcal{A}$, where D is a diagonal matrix with the degree $\deg_i$ of the i-th vertex on the i-th element of the diagonal. While $\mathcal{A}$ is Hermitian, $\mathcal{S}$ is usually not. Still, $\mathcal{S}$ is similar to a Hermitian matrix in the following form: Express $D^{-1/2}\mathcal{A}D^{-1/2} = V\mathbf{A}V^{-1}$ where V is unitary and $\mathbf{A}$ diagonal with real entries (because $D^{-1/2}\mathcal{A}D^{-1/2}$ is Hermitian), then $\mathcal{S} = (D^{-1/2}V)\mathbf{A}(D^{-1/2}V)^{-1}$. Thus, $\mathcal{S}$ has Jordan normal form $W\mathbf{A}W^{-1}$ with $W = D^{-1/2}V$. We see that

$$\begin{aligned}\kappa(W) = \;& \left\|D^{-1/2}V\right\| \cdot \left\|VD^{1/2}\right\| \;\le\; \left\|D^{-1/2}\right\| \left\|D^{1/2}\right\| \|V\| \left\|V^{-1}\right\| \\ \le\;& \sqrt{\frac{\lambda_{\max}(D)}{\lambda_{\min}(D)}} \;\le\; \sqrt{\frac{n}{1}} = \sqrt{n}.\end{aligned}$$

We can therefore always take $\kappa = \sqrt{n}$ in Theorem 24 when we deal with undirected graphs, even when the graph is irregular.

The above discussion shows that $\mathcal{S}$ is similar to the diagonal matrix $\mathbf{A}$ which has a set of real eigenvalues, and therefore so does $\mathcal{S}$. Chung proved that:

▶ **Lemma 26** ([11], Lemma 1.9)**.** *Let $\mathcal{S}$ be a transition matrix of an undirected connected graph with diameter Γ. Then $\gamma'(\mathcal{S}) \geq \frac{1}{\Gamma \cdot \sum_i \deg_i}$.*

Finally, we need the stationary distribution π. However, for an undirected graph $G = (V, E)$ the stationary distribution π is fully explicit and gives weight $\frac{2\deg_i}{|E|}$ to the vertex i. Altogether, we get the theorem for undirected graphs that was stated in the introduction:

▶ **Theorem 27.** *There exists a probabilistic algorithm that gets as input an $n \times n$ stochastic matrix $\mathcal{S}$ that is the transition matrix of an undirected graph and desired accuracy and confidence parameters $\varepsilon, \delta > 0$, outputs a (ε, δ)-approximation of $\mathcal{L}^\star = (\mathcal{I} - \mathcal{S})^\star$ using*

$$O\left(\log \frac{n}{\varepsilon} + \log\log \frac{1}{\delta}\right)$$

space.

We note that the above theorem also holds for *weighted* undirected graphs. To see this, view $\deg_i$ as the sum of weights of the i-th vertex, $\deg_i = \sum_j \mathcal{A}[i,j]$, which is also $\lambda_i(D)$. Then, we can take $\kappa = \sqrt{\lambda_{\max}(D)/\lambda_{\min}(D)}$ in Theorem 24. The stationary distribution is again fully explicit. Finally, analogues of Lemma 26 for weighted undirected graph show that $\gamma'(\mathcal{S})$ is at least inverse-polynomially large in the weights of the graph (e.g., Section 5 in [10]).

When G is undirected we can also approximate in BPL the often used *symmetric normalized Laplacian* , which is

$$\mathcal{L}^{\mathsf{sym}} = \mathcal{I} - D^{-1/2}\mathcal{A}D^{-1/2},$$

where $\mathcal{A}$ is the graph's adjacency matrix and D is the diagonal degrees matrix. We have seen that we can approximate $\mathcal{L}^\star = (\mathcal{I} - D^{-1}\mathcal{A})^\star$ in BPL, and

$$(\mathcal{L}^{\mathsf{sym}})^\star = \left(D^{1/2}\mathcal{L}D^{-1/2}\right)^\star = D^{1/2}\mathcal{L}^\star D^{-1/2}.$$

5.3 On the parameter $\kappa(V)$

Our algorithm's space complexity has a doubly-logarithmic dependency on $\kappa(V)$ – the minimal condition number of all Jordan bases. When the matrix $\mathcal{S}$ has well-separated eigenvalues (namely, the minimal distance between every two eigenvalues is at least polynomially-small), the dependency can be omitted. This is implied by the following theorem:

▶ **Theorem 28** ([32])**.** *Let $\mathcal{A}$ be an $n \times n$ matrix with eigenvalues $\lambda_1, \ldots, \lambda_n$ and suppose $\Delta > 0$ is such that $\min_{i \neq j} |\lambda_i - \lambda_j| \geq \Delta$. Also, let κ_A be the minimal value of $\kappa(V)$ over all V such that $\mathcal{A} = V\mathbf{A}V^{-1}$ is a Jordan decomposition of $\mathcal{A}$. Then, $\kappa_A \leq n \cdot e^{\frac{\|\mathcal{A}\|^2}{2\Delta^2}}$.*

We can thus conclude:

▶ **Theorem 29.** *There exists a probabilistic algorithm that gets as input:*

- *An $n \times n$ irreducible, stochastic matrix $\mathcal{S}$ and a real parameter $\Delta > 0$ so that it is guaranteed that all the eigenvalues of $\mathcal{S}$ are Δ-separated (that is, $|\lambda_i - \lambda_j| \geq \Delta$ for every distinct $\lambda_i, \lambda_j \in \mathsf{Spec}(\mathcal{S})$).*

- *A parameter γ such that $\gamma'(\mathcal{S}) \geq \gamma$.*
- *An approximation $\tilde{\pi}$ of the stationary distribution π of $\mathcal{S}$, where $\|\tilde{\pi} - \pi\| \leq \tau$ and $\tau \leq \frac{\varepsilon}{(T+1)\sqrt{n}}$ for $T = \frac{8n^2}{\gamma^2}\left(1 + \log \frac{n\kappa}{\varepsilon\gamma}\right)$.*

Let $\mathcal{L}$ denote the Laplacian, $\mathcal{L} = \mathcal{I} - \mathcal{S}$. Then, the algorithm outputs a $(3\varepsilon, \delta)$-approximation of $\mathcal{L}^\star$ using

$$O\left(\log \frac{n}{\Delta\gamma\varepsilon} + \log\log \frac{1}{\delta}\right)$$

space.

5.4 Eulerian directed graphs

Eulerian graphs are directed graphs where the in-degree and out-degree of each vertex are the same, and so they generalize both regular directed graphs, and general undirected graphs. The stationary distribution is fully explicit (as in undirected graphs that we mentioned before). In this section we note that for Eulerian graphs γ' is always non-negligible.

▶ **Claim 30.** *Let $\mathcal{S}$ be a transition matrix of a strongly connected Eulerian directed graph with m edges. Then, $\gamma'(\mathcal{S}) \geq \frac{4}{m^2}$.*

Proof. Chung [12] proved that $\gamma'(\mathcal{S})$ is at least the second smallest eigenvalue μ_{n-1} (the smallest eigenvalue is 0) of

$$\mathcal{L}_G^{\mathsf{C}} = I - \frac{\Pi^{1/2}\mathcal{S}\Pi^{-1/2} + \Pi^{-1/2}\mathcal{S}^\dagger\Pi^{1/2}}{2},$$

where Π is a diagonal matrix with the stationary distribution π on the diagonal. Also, in the same paper it is proven that $\mu_{n-1} \geq \frac{4}{m^2}$, which completes the proof. ◀

5.5 Rapidly-mixing graphs

Finally, one way to approximate the stationary distribution is by taking a random walk on G until it converges. This follows directly from Lemma 11 and the fact that $\lim_{k\to\infty} P_G^k = |\mathbf{1}\rangle\langle\pi|$ (see Theorem 3). For undirected graphs (and also Eulerian directed graphs) the walk converges in polynomial time, hence, we can approximate the stationary distribution in logarithmic space, except that there is no need to do that because we have an explicit formula for the stationary distribution anyway.

For general directed graphs (even with bounded degree) the convergence rate can be exponentially small and the approach does not work. Nevertheless, there is a whole class of directed graphs, called *rapidly-mixing graphs*, that converge rapidly even though, usually, there is no explicit formula for the stationary distribution. Clearly, for graphs where the walk converges in polynomial time we can *approximate* the stationary distribution π in logarithmic space.

References

1 Romas Aleliunas, Richard M. Karp, Richard J. Lipton, László Lovász, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *Proceedings of the 20th Annual Symposium on Foundations of Computer Science*, pages 218–223, 1979.

2 Carme Alvarez and Raymond Greenlaw. A compendium of problems complete for symmetric logarithmic space. *Computational Complexity*, 9(2):123–145, 2000.

3 Frank Bauer. Normalized graph laplacians for directed graphs. *Linear Algebra and its Applications*, 436(11):4193–4222, 2012.

4 Adi Ben-Israel and Thomas N. E. Greville. *Generalized inverses: theory and applications*, volume 15. Springer, 2003.

5 Stuart J. Berkowitz. On computing the determinant in small parallel time using a small number of processors. *Information Processing Letters*, 18(3):147–150, 1984. doi:10.1016/0020-0190(84)90018-8.

6 Anders Björner and László Lovász. Chip-firing games on directed graphs. *Journal of Algebraic Combinatorics*, 1(4):305–328, 1992.

7 Allan Borodin, Joachim von zur Gathen, and John E. Hopcroft. Fast parallel matrix and GCD computations. *Information and Control*, 52(3):241–256, 1982. doi:10.1016/S0019-9958(82)90766-5.

8 Richard Bronson. *Matrix Methods: an Introduction*. Gulf Professional Publishing, 1991.

9 William Clough Brown. *A Second Course in Linear Algebra*. Wiley-Interscience, 1988.

10 Fan R. K. Chung. Laplacian of graphs and cheeger's inequalities. *Combinatorics, Paul Erdos is Eighty*, 2(157-172):13–2, 1996.

11 Fan R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

12 Fan R. K. Chung. Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, 9(1):1–19, 2005.

13 Michael B. Cohen, Jonathan Kelner, John Peebles, Richard Peng, Aaron Sidford, and Adrian Vladu. Faster algorithms for computing the stationary distribution, simulating random walks, and more. In *Proceedings of the 57th Annual Symposium on Foundations of Computer Science*, pages 583–592, 2016.

14 Laszlo Csanky. Fast parallel matrix inversion algorithms. *SIAM Journal of Computing*, 5(6):618–623, 1976.

15 James W. Demmel. *Applied numerical linear algebra*. Society for Industrial and Applied Mathematics, 1997.

16 Dean Doron, Amir Sarid, and Amnon Ta-Shma. On approximating the eigenvalues of stochastic matrices in probabilistic logspace. *Computational Complexity*, pages 1–28, 2016.

17 Dean Doron and Amnon Ta-Shma. On the problem of approximating the eigenvalues of undirected graphs in probabilistic logspace. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming*, pages 419–431, 2015.

18 Bill Fefferman and Cedric Yen-Yu Lin. A complete characterization of unitary quantum space. *arXiv preprint arXiv:1604.01384*, 2016.

19 Peter W. Glynn. Upper bounds on poisson tail probabilities. *Operations Research Letters*, 6(1):9–14, 1987.

20 Chris Godsil. Eigenvalues of graphs and digraphs. *Linear Algebra and its Applications*, 46:43–50, 1982.

21 Chris Godsil and Gordon F. Royle. *Algebraic Graph Theory*. Springer, 2013.

22 Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, USA, 3rd edition, 1996.

23 Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. Society for Industrial and Applied Mathematics, 2008.

24 Jonathan A. Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. A simple, combinatorial algorithm for solving SDD systems in nearly-linear time. In *Proceedings of the 45th Annual Symposium on Theory of Computing*, pages 911–920, 2013. doi:10.1145/2488608.2488724.

25 Adam R. Klivans and Dieter Van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

26 Harry R. Lewis and Christos H. Papadimitriou. Symmetric space-bounded computation. *Theoretical Computer Science*, 19(2):161–187, 1982.

27 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

28 Noam Nisan. RL⊆SC. *Computational Complexity*, 4(1):1–11, 1994.

29 Richard Peng and Daniel A. Spielman. An efficient parallel solver for SDD linear systems. In *Proceedings of the 46th Annual Symposium on Theory of Computing*, pages 333–342, 2014. doi:10.1145/2591796.2591832.

30 Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM*, 55(4), 2008.

31 Michael E. Saks and Shiyu Zhou. $\mathrm{BP_H SPACE(S)} \subseteq \mathrm{DSPACE(S^{3/2})}$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999.

32 Russell A. Smith. The condition numbers of the matrix eigenvalue problem. *Numerische Mathematik*, 10(3):232–240, 1967.

33 Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the 36th Annual Symposium on Theory of Computing*, pages 81–90, 2004. doi:10.1145/1007352.1007372.

34 Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025, 2011. doi:10.1137/08074489X.

35 Daniel A. Spielman and Shang-Hua Teng. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM Journal on Computing*, 42(1):1–26, 2013. doi:10.1137/080744888.

36 Daniel A. Spielman and Shang-Hua Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM Journal on Matrix Analysis and Applications*, 35(3):835–885, 2014. doi:10.1137/090771430.

37 Amnon Ta-Shma. Inverting well conditioned matrices in quantum logspace. In *Proceedings of the 45th Annual Symposium on Theory of Computing*, pages 881–890, 2013. doi:10.1145/2488608.2488720.

38 Vladimir Trifonov. An $O(\log n \log\log n)$ space algorithm for undirected st-connectivity. *SIAM Journal on Computing*, 38(2):449–483, 2008.

39 Nisheeth K. Vishnoi. *$Lx = b$ – Laplacian Solvers and their Algorithmic Applications.* Now publishers, 2013.

40 John Watrous. Space-bounded quantum complexity. *Journal of Computer and System Sciences*, 59(2):281–326, 1999.

Streaming Periodicity with Mismatches*†

Funda Ergün[1], Elena Grigorescu[2], Erfan Sadeqi Azer[3], and Samson Zhou[4]

1 **School of Informatics and Computing, Indiana University, Bloomington, IN, USA**
fergun@indiana.edu

2 **Department of Computer Science, Purdue University, West Lafayette, IN, USA**
elena-g@purdue.edu

3 **School of Informatics and Computing, Indiana University, Bloomington, IN, USA**
esadeqia@indiana.edu

4 **Department of Computer Science, Purdue University, West Lafayette, IN, USA**
samsonzhou@gmail.com

Abstract

We study the problem of finding all k-periods of a length-n string S, presented as a data stream. S is said to have k-period p if its prefix of length $n-p$ differs from its suffix of length $n-p$ in at most k locations.

We give a one-pass streaming algorithm that computes the k-periods of a string S using $\mathrm{poly}(k, \log n)$ bits of space, for k-periods of length at most $\frac{n}{2}$. We also present a two-pass streaming algorithm that computes k-periods of S using $\mathrm{poly}(k, \log n)$ bits of space, regardless of period length. We complement these results with comparable lower bounds.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases String algorithms, Streaming algorithms, Pattern matching, Randomized algorithms, Sublinear algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.42

1 Introduction

In this paper we are interested in finding (possibly imperfect) periodic trends in sequences given as streams. Informally, a sequence is said to be *periodic* if it consists of repetitions of a block of characters; e.g., *abcabcabc* consists of repetitions of *abc*, of length 3, and thus has period 3. The study of periodic patterns in sequences is valuable in fields such as string algorithms, time series data mining, and computational biology. The question of finding the smallest period of a string is a fundamental building block for many string algorithms, especially in pattern matching, such as the classic Knuth-Morris-Pratt [21] algorithm. The general technique for many pattern matching algorithms is to find the periods of prefixes of the pattern in a preprocessing stage, then use them as a guide for ruling out locations where the pattern cannot occur, thus improving efficiency.

[*] A full version of the paper is available at http://homes.soic.indiana.edu/fergun/PUBLICATIONS/mismatchperiodicity.pdf.

[†] Funda Ergün's research is supported by NSF CCF-1619081; Elena Grigorescu's and Samson Zhou's research is supported by NSF CCF-1649515.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 42; pp. 42:1–42:21
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

While finding exact periods is fundamental to pattern matching, in real life, it is unrealistic to expect data to be perfectly periodic. In this paper, we assume that even when there is a fixed period, data might subtly change over time. In particular, we might see *mismatches,* defined as locations in the sequence where a block is not the same as the previous block. For instance, while *abababababab* is perfectly periodic, *abababacacac* contains one mismatch where *ab* becomes (and stays) *ac*. This model captures periodic events that undergo permanent modifications over time (e.g., statistics that remain generally cyclic but experience infrequent permanent changes or errors). We consider our problem in the *streaming* setting, where the input is received in a sequential manner, and is processed using sublinear space.

Our problem generalizes exact periodicity studied in [12], where the authors give a one-pass, $\mathcal{O}\left(\log^2 n\right)$-space algorithm for finding the smallest *exact* period of stream S of length n, when the period is at most $n/2$, as well as a linear space lower bound when the period is longer than $n/2$. They use two standard and equivalent definitions of periodicity: S has period p if it is of the form $B^\ell B'$ where B is a block of length p that appears $\ell \geq 1$ times in a row, and B' is a prefix of B. For instance, *abcabcabcab* has period 3 where $B = abc$, and $B' = ab$. Equivalently, the length $n-p$ prefix of S is identical to its length $n-p$ suffix. These definitions imply that at most k of the repeating blocks differ from the preceding ones. According to this definition, for instance, *abcabdabdae* is 2-periodic with period 3, with the mismatches occurring at positions 6 and 11.

In order to allow mismatches in S while looking for periodicity in small space, we utilize the fingerprint data structure introduced for pattern matching with mismatches by [25, 7]. Ideally, one would hope to combine results from [12] and [7] to readily obtain an algorithm for detecting k-periodicity. Unfortunately, reasonably direct combinations of these techniques do not seem to work. This is due to the fact that, in the presence of mismatches, the essential structural properties of periods break down. For instance, in the exact setting, if S has periods p and q, it must also have period r, where r is any positive multiple of p or q. It must also have period $d = gcd(p, q)$. These are not necessarily true when there are mismatches; as an example consider the following.

▶ **Example 1.** $S = aaaaba$ has only one mismatch where $S[i] \neq S[i+2]$ (over all non range-violating values of i); likewise where $S[i] \neq S[i+3]$, thus S is 1-periodic with periods 2 and 3. S is *not* 1-periodic with period $1 = \mathsf{gcd}\,(2,3)$ as it has *two* mismatches where $S[i] \neq S[i+1]$.

In the exact setting the smallest period t determines the entire structure of S as all other periods must be multiples of t. This property does not necessarily hold when we allow mismatches, thus the smallest period does not carry as much information as in the exact case. Similarly, overlaps of a pattern with itself in S exhibits a much less well-defined periodic structure in the presence of mismatches. This makes it much harder to achieve the fundamental space reduction achievable in exact periodicity computation, where this kind of structure is crucially exploited.

1.1 Our Results

Given the structural challenges introduced by the presence of mismatches, we first focus on understanding the unique structural properties of k-periods and the relationship between the period p, and the number of mismatches k (See Theorem 9). This understanding gives us tools for "compressing" our data into sublinear space. We proceed to present the following on a given stream S of length n:

1. a two-pass streaming algorithm that computes all k-periods of S using $\mathcal{O}\left(k^4 \log^9 n\right)$ space, *regardless of period length* (see Section 4)
2. a one-pass streaming algorithm that computes all k-periods of length at most $n/2$ of S using $\mathcal{O}\left(k^4 \log^9 n\right)$ space (see Section 5)
3. a lower bound that any one-pass streaming algorithm that computes all k-periods of S requires $\Omega(n)$ space (see Section 6)
4. a lower bound that for $k = o(\sqrt{n})$ with $k > 2$, any one-pass streaming algorithm that computes all k-periods of S with probability at least $1 - 1/n$ requires $\Omega(k \log n)$ space, even under the promise that the k-periods are of length at most $n/2$. (see Section 6)

Given the above results, it is trivial to modify the algorithms to return, rather than all k-periods, the smallest, largest, or any particular k-period of S.

1.2 Related Work

Our work extends two natural directions in sublinear algorithms for strings: on one hand the study of the repetitive structure of long strings, and on the other hand the notion of approximate matching of patterns, in which the algorithm can detect a pattern even when some of it got corrupted.

In the first line of work, Ergün *et al.* [12] initiate the study of streaming algorithms for detecting the period of a string, using $poly(\log n)$ bits of space. Indyk *et al.* [19] also studied mining periodic patterns in streams, [10] studied periodicity in time-series databases and online data, and Crouch and McGregor [9] study periodicity via linear sketches. [13] and [23] studied the problem of distinguishing periodic strings from aperiodic ones in the property testing model of sublinear-time computation. Furthermore, [1] studied approximate periodicity in RAM model under the Hamming and swap distance metrics.

The pattern matching literature is a vast area (see [3] for a survey) with many variants. Following the pattern matching streaming algorithm of Porat and Porat [25], Clifford *et al.* [7] recently show improved streaming algorithms for the k-mismatch problem, as well as offline and online variants. We adapt the use of sketches from [7] though there are some other works with different sketches for strings ([2], [5], [27] and [26]). [8] also showed several lower bounds for online pattern matching problem.

This line of work is also related to the detection of other natural patterns in strings, such as palindromes or near palindromes. Ergün *et al.* [4] initiate the study of this problem and give sublinear-space algorithms, while [16] show lower bounds. In recent work, [18] extend this problem to finding near-palindromes (i.e., palindromes with possibly a few corrupted entries).

Many ideas used in these sublinear algorithms stem from related work in the classical offline model. The well-known KMP algorithm [22] initially used periodic structures to search for patterns within a text. Galil *et al.*[14] later improved the space performance of this pattern matching algorithm. Recently, [15] also used the properties of periodic strings for pattern matching when the strings are compressed. These interesting properties have allowed several algorithms to satisfy some non-trivial requirements of respective models (see [17], [6] for example).

2 Preliminaries

We assume our input is a stream $S[1,\ldots,n]$ of length $|S| = n$ over some alphabet Σ. The i^{th} character of S is denoted $S[i]$, and the substring between locations i and j (inclusive) $S[i,j]$. Two strings $S, T \in \Sigma^n$ are said to have a *mismatch* at index i if $S[i] \neq T[i]$, and their Hamming distance is the number of such mismatches, denoted $\mathsf{HAM}(S,T) = \Big|\{i \mid S[i] \neq T[i]\}\Big|$. We denote the concatenation of S and T by $S \circ T$.

S is said to have *period* p if $S[x] = S[x+p]$ for all $1 \leq x \leq n-p$; more succinctly, if $S[1,n-p] = S[p+1,n]$. In general, we say S has k-period p (i.e., S has period p with k mismatches) if $S[x] = S[x+p]$ for all but at most k valid indices x. Equivalently, S has k-period p if and only if $\mathsf{HAM}(S[1,n-p], S[p+1,n]) \leq k$.

▶ **Observation 2.** *If p is a k-period of S, then at most k of the sequence of substrings $S[1,p], S[p+1,2p], S[2p+1,3p], \ldots$ can differ from the previous substring in the sequence.*

When obvious from the context, given k-period p, we denote as a *mismatch* a position i for which $S[i] \neq S[i+p]$.

▶ **Example 3.** The string $S = aaaaaabbccd$ has 3-period equal to 1, since $S[i] = S[i+1]$ for all valid locations i except mismatches at $i = 6, 8, 10$. On the other hand, $S = abcabcadcabc$ has 2-period equal to 3 since $S[i] = S[i+3]$ for all valid i except mismatches $i = 5, 8$.

The following observation notes that the number of mismatches between two strings is an upper bound on the number of mismatches between their prefixes of equal length.

▶ **Observation 4.** *If p is a k-period of S, then for any $x \leq n-p$, the number of mismatches between $S[1,x]$ and $S[p+1,p+x]$ is at most k.*

Given two integers x and y, we denote their greatest common divisor by $\mathsf{gcd}(x,y)$.

We repeatedly use data structures and subroutines that use Karp-Rabin fingerprints. For more about the properties of Karp-Rabin fingerprints see [20], but for our purposes, the following suffice:

▶ **Theorem 5** ([7]). *Given two strings S and T of length n, there exists a data structure that uses $\mathcal{O}\left(k \log^6 n\right)$ bits of space, and outputs whether $\mathsf{HAM}(S,T) > k$ or $\mathsf{HAM}(S,T) \leq k$, along with the set of locations of the mismatches in the latter case.*

From here, we use the term *fingerprint* to refer to this data structure.

2.1 The k-Mismatch Algorithm

For our string-matching tasks, we utilize an algorithm from [7], whose parameters are given in Theorem 6. For us, string matching is a tool rather than a goal; as a result, we require additional properties from the algorithm that are not obvious at first glance. In Corollary 7 we consider these properties. Throughout our algorithms and proofs, we frequently refer to this algorithm as the *k-Mismatch Algorithm.*

▶ **Theorem 6** ([7]). *Given a pattern P of length ℓ, a text T of length n and some mismatch threshold k, there exists an algorithm that, with probability $1 - \frac{1}{n^2}$, outputs all indices i such that $\mathsf{HAM}(T[i, i+\ell-1], P) \leq k$ using $\mathcal{O}\left(k^2 \log^8 n\right)$ bits of space.*

Whereas the pattern in the k-Mismatch Algorithm is given in advance and can be preprocessed before the text, in our case the pattern is a prefix of the text, and the algorithm must return any matches of this pattern, starting possibly at location 2, well within the original occurrence of the pattern itself. (Consider text ‘abcdabcdabcdabcd’ and the pattern ‘abcdabcd,’ the first six characters of the text. The first match starts at location 4, but the algorithm does not finish reading the full pattern until it has read location 6.) To eliminate a potential problem due to this requirement, we make modifications so that the algorithm can search for all matches in S of a prefix of S.

▶ **Corollary 7.** *Given a string S and an index x, there exists an algorithm which, with probability $1 - \frac{1}{n^2}$, outputs all indices i where* $\mathsf{HAM}\left(S[1,x], S[i+1, i+x]\right) \leq k$ *using* $\mathcal{O}\left(k^2 \log^8 n\right)$ *bits of space.*

Proof. We claim that the algorithm of Theorem 6 can be arranged and modified to output all such indices i. We need to input $S[1,x]$ as the pattern and $S[2,n]$ as the text for this algorithm.

Thus, it suffices to argue that the data structure for the pattern is built in an online fashion. That is, after reading each symbol of the pattern, the data structure corresponding to the prefix of the pattern that has already been read is updated and ready to use. Moreover, the process of building the data structure for the text should not depend on the pattern. The only dependency between these two processes can be that they need to use the same randomness. Therefore, the algorithm only needs to decide the randomness before starting to process the input and share it between processes.

The algorithm of Theorem 6 has a few components, explained in the proof of Theorem 1.2 in [7]. Here, we go through these components and explain how they satisfy the conditions we mentioned.

The main data structure for this algorithm is also used in Theorem 5. In this data structure, each symbol is partitioned to various subpatterns determined by the index of the symbol along with predetermined random primes. Each subpattern is then fed to a dictionary matching algorithm. The dictionary entries are exactly the subpatterns of the original patterns and thus can be updated online.

The algorithm also needs to consider run-length encoding for each of these subpatterns in case they are highly periodic. It is clear that run-length encoding can be done independently for the pattern and the text.

Finally the approximation algorithm (Theorem 1.3 of [7]) uses a similar data structure to Theorem 5, but with different magnitudes for primes. Thus, the entire algorithm can be modified to run in an online fashion. ◀

3 Our Approach

Our approach to find all the k-periods of S is to first determine a set $\mathcal{T}$ of candidate k-periods, which is guaranteed to be a superset of all the true k-periods. We first describe the algorithm to find the k-period in two passes. In the first pass, we let $\mathcal{T}$ be the set of indices π that satisfy

$$\mathsf{HAM}\left(S[1,x], S[\pi+1, \pi+x]\right) \leq k,$$

for some appropriate value of x that we specify later. Note that by Observation 4, all k-periods must satisfy the above inequality. We show that even though $\mathcal{T}$ may be linear in size, we can succinctly represent $\mathcal{T}$ by adding a few additional indices into $\mathcal{T}$. We then show how to use the compressed version of $\mathcal{T}$ during the second pass to verify the candidates and output the true k-periods of S.

This strategy does not work if we are allowed only one pass; by the time we discover a candidate k-period p, it may be too late for us to start collecting the extra data needed to verify p (in the two-pass version this is not a problem, as the extra pass allows us to go back to the start of S and any needed data). We approach this problem by utilizing a trick from [12] of identifying candidate periods p using non-uniform criteria depending on the value of p. Using this idea, once a candidate period is found, it is not too late to verify that it is a true k-period, and the data can still be compressed into sublinear size.

Perhaps the biggest hidden challenge in the above approach is due to the major structural differences between exactly periodic and k-periodic strings; k-periodic strings show much less structure than exactly periodic strings. As a result, incremental adaptations of existing techniques on periodic strings do not yield corresponding schemes for k-periodic strings. In order to achieve small space, one needs to explore the weaker structural properties of k-periodic streams. A large part of the effort in this work is in formalizing said structure (see Appendix A), culminating in Theorem 23 and its proof, as well as exploring its application to our algorithms.

To show lower bounds for randomized algorithms finding the smallest k-period, we use a strategy similar to that in [12], using a reduction from the Augmented Index Problem. To show lower bounds for randomized algorithms finding the smallest k-period given the promise that the smallest k-period is at most $\frac{n}{2}$, we use Yao's Principle [28].

4 Two-Pass Algorithm to Compute k-Periods

In this section, we provide a two-pass, $\mathcal{O}\left(k^4 \log^9 n\right)$-space algorithm to output all k-periods of S. The general approach is to first identify a superset of the k-periods of S, based on the self-similarity of S, detected via the k-Mismatch algorithm of [7] as a black box. Unfortunately, while this tool allows us to match parts of S to each other, we get only incomplete information about possible periods, and this information is not readily stored in small space due to insufficient structure. We explore the structure of periods with mismatches in order to come up with a technique that massages our data into a form that can be compressed in small space, and is easily uncompressed. During the second pass, we go over S as well as the compressed data to verify the candidate periods.

We consider two classes of periods by their length, and run two separate algorithms in parallel. The first algorithm identifies all k-periods p with $p \le \frac{n}{2}$, while the second algorithm identifies all k-periods p with $p > \frac{n}{2}$.

4.1 Finding small k-periods

Our algorithm for finding periods of length at most $n/2$ proceeds in two passes. In the first pass, we identify a set $\mathcal{T}$ of candidate k-periods, and formulate its compressed representation, $\mathcal{T}^C$. In the second pass, we recover each index from $\mathcal{T}^C$ and verify whether or not it is a k-period. We need $\mathcal{T}$ and $\mathcal{T}^C$ to satisfy four properties.

1. All true k-periods (likely accompanied by some candidate k-periods that are false positives) are in $\mathcal{T}$.
2. $\mathcal{T}^C$ can be stored in sublinear space.
3. $\mathcal{T}$ can be fully recovered from $\mathcal{T}^C$ in small space.
4. The verification process in the second pass weeds out those candidates that are not true periods in sublinear space.

We now describe our approach and show how it satisfies the above properties.

4.2 Pass 1: Property 1

We crucially observe that any k-period p must satisfy the requirement

$$\mathsf{HAM}\left(S[1, x], S[p+1, p+x]\right) \le k$$

for all $x \le n - p$, and specifically for $x = \frac{n}{2}$. This observation allows us to refer to indices as periods, as the index $p + 1$ where the requirement is satisfied corresponds to (possible)

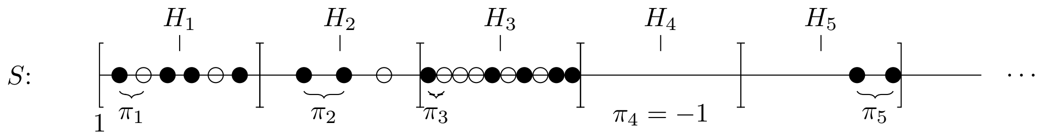


Figure 1 Observe that all dots in each interval are equally spaced after the first. These dots represent $\mathcal{T}^c$: the black dots represent $\mathcal{T}$, while the white dots are added to convert the irregularly spaced black dots into regularly spaced dot sequences.

k-period p. For the remainder of this algorithm, we set $x = \frac{n}{2}$, and designate the indices $p+1$ that satisfy the requirement with $x = \frac{n}{2}$ as candidate k-periods; collectively these indices serve as $\mathcal{T}$. Since satisfying this requirement is necessary but not sufficient for a candidate to be a real k-period, Property 1 follows.

4.3 Pass 1: Property 2

Observe that $\mathcal{T}$ could be linear in size, so we cannot store each index explicitly. We observe that if our indices followed an arithmetic progression, they could be kept implicitly in very succinct format (as is the case where there are no mismatches). Unfortunately, due to the presence of mismatches in S, such a regular structure does not happen. However, we show that it is still possible to implicitly add a small number of extra indices to our candidates and end up with an arithmetic series and allow for succinct representation. Our algorithm produces several such series, and represents each one in terms of its first index and the increment between consecutive terms, obtaining $\mathcal{T}^C$ from $\mathcal{T}$, with the details given below.

In order to compress $\mathcal{T}$ into $\mathcal{T}^C$, we partition $[1, x]$ into the $2mk+2$ disjoint intervals $H_j = \left[\frac{jx}{2(mk+1)} + 1, \frac{(j+1)x}{2(mk+1)}\right)$, where $m = \log n$. The goal is, possibly through the addition of extra candidates, to represent the candidates in each interval as a single arithmetic series. This series will be represented by its first term, as well as the increment between its consecutive terms, π_j. As each new candidate arrives, we update π_j (except for the first update, π_j never increases, and it may shrink by an integer factor). Throughout the process, we maintain the invariant, by updating π_j, that the arithmetic sequence represented in H_j contains all candidates in H_j output by the k-Mismatch algorithm. Then it is clear that $\mathcal{T}^C$ and $\{\pi_j\}$ take sublinear space, satisfying Property 2.

4.4 Pass 1: Property 3

It remains to describe how to update π_j. The first time we see two candidates in H_j, we set π_j to be the increment between the candidates (before, it is set to -1). Each subsequent time we see a new candidate index in the interval H_j, we update π_j to be the greatest common divisor of π_j and the increment between the candidate and the smallest index in $\mathcal{T} \cap H_j$, which is kept explicitly. For instance, if our first candidate index is 10, and afterwards we receive 22, 26, 32 (assume the interval ends at 35), our π_j values over time are -1, 12, 4, 2. Ultimately, the candidates that we will be checking in Pass 2 will be 10, 12, 14, 16, 18, ..., 34. For another example, see Figure 1.

We now need to show that the above invariant is maintained throughout the algorithm. To do this, we show that any k-period $p \in H_j$ is an increment of some multiple of π_j away from the smallest index in $\mathcal{T} \cap H_j$. Then, if we insert implicitly into $\mathcal{T}$ *all indices* in H_j whose distance from the smallest index in $\mathcal{T} \cap H_j$ is a multiple of π_j, we will guarantee that any k-period in H_j will be included in $\mathcal{T}$.

We now show that any k-period p is implicitly represented in, and can be recovered from $\mathcal{T}^C$ and the values $\{\pi_j\}$ at the end of the first pass.

▶ **Lemma 8.** *If $p < \frac{n}{2}$ is a k-period and $p \in H_j$, then p can be recovered from $\mathcal{T}^C$ and π_j.*

Proof. Since $p \in H_j$ is a k-period, then it satisfies $\mathsf{HAM}\left(S[1, n-p], S[p+1, n]\right) \leq k$. More specifically, $i = p$ satisfies

$$\mathsf{HAM}\left(S\left[1, \frac{n}{2}\right], S\left[i+1, \frac{n}{2}+i\right]\right) \leq k$$

and will be reported by the k-Mismatch Algorithm. If there is no other index in $\mathcal{T}^C \cap H_j$, then p will be inserted into $\mathcal{T}^C$ in the first pass, so p can clearly be recovered from $\mathcal{T}^C$.

On the other hand, if there is another index q in $\mathcal{T}^C \cap H_j$, then π_j will be updated to be a divisor of the pairwise distances. Hence, the increment $p - q$ is a multiple of π_j. Any change that might later happen to π_j will be due to a gcd operation, and thus, will reduce it by a factor by at least 2. Thus, $p - q$ will remain a multiple of the final value of π_j, and p will be recovered at the end of the first pass as a member of $\mathcal{T}$. ◀

Thus Property 3 is satisfied. The first pass algorithm in full appears below.

```
(To determine any $k$-period $p$ with $p \leq \frac{n}{2}$):
First pass:
1. Initialize $\pi_j = -1$ for each $0 \leq j < 2k \log n + 2$.
2. Initialize $\mathcal{T}^C = \emptyset$.
3. For each index $i$ such that (using the $k$-Mismatch algorithm)
       $\mathsf{HAM}\left(S\left[1, \frac{n}{2}\right], S\left[i+1, \frac{n}{2}+i\right]\right) \leq k$.
   - For the integer $j$ for which $i$ is in the interval $H_j = \left[\frac{jn}{4(k \log n+1)} + 1, \frac{(j+1)n}{4(k \log n+1)}\right)$:
     a. If there exists no candidate $t \in \mathcal{T}^C$ in the interval $H_j$, then add $i$ to $\mathcal{T}^C$.
     b. Otherwise, let $t$ be the smallest candidate in $\mathcal{T}^C$ and either $\pi_j = -1$ or $\pi_j > 0$.
        If $\pi_j = -1$, then set $\pi_j = i - t$. Otherwise, set $\pi_j = \mathsf{gcd}\left(\pi_j, i - t\right)$.
```

4.5 Pass 2: Property 4

Our task in the second pass is to verify whether each candidate recovered from $\mathcal{T}^C$ and $\{\pi_j\}$ is actually a k-period or not. Thus, we must simultaneously check whether $\mathsf{HAM}\left(S[1, n-p], S[p+1, n]\right) \leq k$ for each candidate p, without using linear space. Fortunately, Theorem 9 states that at most $32k^2 \log n + 1$ unique fingerprints for substrings of length π_j are sufficient to recover the fingerprints of both $S[1, n-p]$ and $S[p+1, n]$ for any $p \in H_j$.

Before detailing, we first state a structural property, whose proof we defer to Appendix A. This property states that the greatest common divisor of the pairwise difference of any candidate k-periods within H_j must be a $(32k^2 \log n + 1)$-period.

▶ **Theorem 9.** *For some $0 \leq j < 2mk + 2$, let*

$$\mathcal{I}_j = \left\{i \in H_j \mid \mathsf{HAM}\left(S[1, x], S[i+1, i+x]\right) \leq k\right\}.$$

For any $p_1 < \ldots < p_m \in \mathcal{I}$, the greatest common divisor d of $p_2 - p_1, p_3 - p_1 \ldots, p_m - p_1$ satisfies

$$\mathsf{HAM}\,(S[1,x], S[d+1, d+x]) \leq 32mk^2 + 1.$$

Observe that π_j is exactly d. Moreover, each time the value of π_j changes, it gets divided by an integer factor at least equal to 2, ending up finally as a positive integer. Since $\pi_j \leq n$, this change can occur at most $\log n$ times, and so $m \leq \log n$. We now show that we can verify all candidates in sublinear space.

▶ **Lemma 10.** *Let p_i be a candidate k-period for a string S, with $p_1 < p_2 < \ldots < p_m$ all contained within H_j. Given the fingerprints of $S[1, n - p_1]$ and $S[p_1 + 1, n]$, we can determine whether or not S has k-period p_i for any $1 \leq i \leq m$ by storing at most $32k^2 \log n + 1$ additional fingerprints.*

Proof. Consider a decomposition of S into substrings w_i of length p_i, so that $S = w_1 \circ w_2 \circ w_3 \circ \ldots$. Note that each index i for which $w_i \neq w_{i+1}$ corresponds with at least one mismatch. It follows from Observation 2 that there exist at most k indices i for which $w_i \neq w_{i+1}$. Thus, recording the fingerprints and locations of these indices i suffice to determine whether or not there are k mismatches for candidate period p_i.

By Theorem 9, the greatest common divisor of the difference between each term in $\mathcal{I}$ is a $(32k^2 \log n + 1)$-period π_j. Thus, S can be decomposed $S = v \circ v_1 \circ v_2 \circ v_3 \circ \ldots$ so that v has length p_1, and each substring v_i has length π_j. It follows from Observation 2 that there exist at most $32k^2 \log n + 1$ indices i for which $v_i \neq v_{i+1}$. Therefore, recording the fingerprints and locations of these indices i allow us to recover the fingerprint of $S[1, n - p_i]$ from the fingerprint of $S[1, n - p_{i-1}]$, since $p_i - p_{i-1}$ is a multiple of π_j. Similarly, we can recover the fingerprint of $S[p_i + 1, n]$ from the fingerprint of $S[p_{i-1} + 1, n]$. Hence, we can confirm whether or not p_i is a k-period. ◀

The second pass algorithm in full follows.

```
(To determine all the $k$-periods $p$ with $p \leq \frac{n}{2}$):
Second pass:
1. For each $t$ such that $t \in \mathcal{T}^C$:
    a. Let $j$ be the integer for which $t$ is in the interval $H_j = \left[\frac{jn}{4(k \log n+1)} + 1, \frac{(j+1)n}{4(k \log n+1)}\right)$
    b. If $\pi_j > 0$, then record up to $32k^2 \log n + 1$ unique fingerprints of length $\pi_j$ and of
       length $t$, starting from $t$.
    c. Otherwise, record up to $32k^2 \log n + 1$ unique fingerprints of length $t$, starting
       from $t$.
    d. Check if $\mathsf{HAM}\,(S[1, n - t], S[t + 1, n]) \leq k$ and return $t$ if this is true.
2. For each $t$ which is in interval $H_j = \left[\frac{jn}{4(k \log n+1)} + 1, \frac{(j+1)n}{4(k \log n+1)}\right)$ for some integer $j$:
    - If there exists an index in $\mathcal{T}^C \cap H_j$ whose distance from $t$ is a multiple of $\pi_j$, then
      check if $\mathsf{HAM}\,(S[1, n - t], S[t + 1, n]) \leq k$ and return $t$ if this is true.
```

This proves Property 4. Next, we show the correctness of the algorithm for small k-periods.

▶ **Lemma 11.** *For any k-period $p \leq \frac{n}{2}$, the algorithm outputs p.*

Proof. Since the intervals $\{H_j\}$ cover $\left[1, \frac{n}{2}\right]$, then $p \in H_j$ for some j. It follows from Lemma 8 that after the first pass, p can be recovered from $\mathcal{T}$ and π_j. Thus, the second pass tests whether or not p is a k-period. By Lemma 10, the algorithm outputs p, as desired. ◀

4.6 Finding large k-periods

As in the previous discussion, we would like to pick candidate periods during our first pass. However, if a k-period p satisfies $p > \frac{n}{2}$, then clearly it will no longer satisfy

$$\mathsf{HAM}\left(S\left[1,\frac{n}{2}\right],S\left[p+1,p+\frac{n}{2}\right]\right)\le k,$$

as $p+\frac{n}{2} > n$, and $S\left[p+\frac{n}{2}\right]$ is undefined. Instead, recall that $\mathsf{HAM}\left(S[1,x] = S[p+1,p+x]\right) \le k$ for all $x \le n-p$. Ideally, when choosing candidate periods p based on their satisfying this formula, we would like to use as large an x as possible without exceeding $n-p$, but we cannot do this without knowing the value of p. Instead, [12] observes we can try exponentially decreasing values of x: we run $\log n$ instances of the algorithm sequentially, with $x = \frac{n}{2}, \frac{n}{4}, \ldots$, since one of these values of x must be the largest one that does not lead to an illegal index of S. Therefore, the desired instance produces p, while all other instances do not.

(To determine a k-period p if $p > \frac{n}{2}$):

First pass:

1. Initialize $\pi_j^{(m)} = -1$ for each $0 \le j < 2k\log n + 2$ and $0 \le m \le \log n$.
2. Initialize $\mathcal{T}_m^C = \emptyset$.
3. For each index i, let r be the largest m such that $\frac{n}{2}+\frac{n}{4}+\ldots+\frac{n}{2^r} \le i$. Using the k-Mismatch algorithm, check whether

   $$\mathsf{HAM}\left(S\left[1,\frac{n}{2^r}\right],S\left[i+1,i+\frac{n}{2^r}\right]\right)\le k.$$

   If so, let $R = \frac{n}{2}+\frac{n}{4}+\ldots+\frac{n}{2^r-1}$ and j be the integer for which i is in the interval

   $$H_j^{(r)} = \left[R+\frac{nj}{2^{r+1}(k\log n+1)}+1, R+\frac{n(j+1)}{2^{r+1}(k\log n+1)}\right)$$

   a. If there exists no candidate $t \in \mathcal{T}_r^C$ in the interval $H_j^{(r)}$, then add i to $\mathcal{T}_r^C$.
   b. Otherwise, let t be the smallest candidate in $\mathcal{T}_r^C$ and either $\pi_j^{(r)} = -1$ or $\pi_j^{(r)} > 0$. If $\pi_j^{(r)} = -1$, then set $\pi_j^{(r)} = i-t$. Otherwise, set $\pi_j^{(r)} = \mathsf{gcd}\left(\pi_j^{(r)}, i-t\right)$.

This partition of $[1,n]$ into the disjoint intervals $\left[1,\frac{n}{2}\right]$, $\left[\frac{n}{2}+1,\frac{n}{2}+\frac{n}{4}\right]$, ... guarantees that any k-period p is contained in one of these intervals. Moreover, the intervals $\{H_j^{(r)}\}$ partition

$$\left[\frac{n}{2}+\frac{n}{4}+\ldots+\frac{n}{2^{r-1}},\frac{n}{2}+\ldots+\frac{n}{2^r}\right],$$

and so p can be recovered from $\mathcal{T}_r^C$ and $\{\pi_j^{(r)}\}$. We now present the algorithm for the second-pass to find all k-periods p for which $p > \frac{n}{2}$.

Second pass:

1. For each t and any r such that $t \in \mathcal{T}_r^C$:
   - a. Let $R = \frac{n}{2} + \frac{n}{4} + \ldots + \frac{n}{2^{r-1}}$ and j be the integer for which t is in the interval
   $$H_j^{(r)} = \left[R + \frac{nj}{2^{r+1}(k\log n + 1)} + 1, R + \frac{n(j+1)}{2^{r+1}(k\log n + 1)}\right)$$
   - b. If $\pi_j^{(r)} > 0$, then record up to $32k^2\log n + 1$ unique fingerprints of length $\pi_j^{(r)}$ and of length t, starting from t.
   - c. Otherwise, record up to $32k^2\log n + 1$ unique fingerprints of length t, starting from t.
   - d. Check if $\mathsf{HAM}\left(S[1, n-t], S[t+1, n]\right) \le k$ and return t if this is true.
2. For each t which is in interval $H_j^{(r)} = \left[R + \frac{nj}{2^{r+1}(k\log n+1)} + 1, R + \frac{n(j+1)}{2^{r+1}(k\log n+1)}\right)$ for some integer j:
   - a. If there exists an index in $\mathcal{T}_r^C \cap H_j^{(r)}$ whose distance from t is a multiple of $\pi_j^{(r)}$, then check if $\mathsf{HAM}\left(S[1, n-t], S[t+1, n]\right) \le k$ and return t if this is true.

Since correctness follows from the same arguments as the case where $p \le \frac{n}{2}$, it remains to analyze the space complexity of our algorithm.

▶ **Theorem 12.** *There exists a two-pass algorithm that outputs all the k-periods of a given string using $\mathcal{O}\left(k^4 \log^9 n\right)$ space.*

Proof. In the first pass, for each $\mathcal{T}_m$, we maintain a k-Mismatch algorithm which requires $\mathcal{O}\left(k^2\log^8 n\right)$ bits of space, as in Corollary 7. Since $1 \le m \le \log n$, we require $\mathcal{O}\left(k^2\log^9 n\right)$ bits of space in total. In the second pass, we keep up to $\mathcal{O}\left(k^2\log n\right)$ fingerprints for any set of indices in $\mathcal{T}_m$. Each fingerprint requires space $\mathcal{O}\left(k\log^6 n\right)$ and there may be $\mathcal{O}\left(k\log n\right)$ indices in $\mathcal{T}_m$ for each $1 \le m \le \log n$, for a total of $\mathcal{O}\left(k^4\log^7 n\right)$ bits of space. Thus, $\mathcal{O}\left(k^4\log^9 n\right)$ bits of space suffice for both passes. ◀

5 One-Pass Algorithm to Compute k-Periods

We now give a one-pass algorithm that outputs all the k-periods smaller than $\frac{n}{2}$. Similar to two-pass algorithm, we have two processes running in parallel. The first process handles all the k-periods p with $p \le \frac{n}{4}$, while the second process handles the k-periods p with $p > \frac{n}{4}$. Both processes are designed again based on the crucial observation that all the k-periods p must satisfy $\mathsf{HAM}\left(S[1, x], S[p+1, p+x]\right) \le k$ for all $x \le n-p$. In the first process, we set $x = \frac{n}{2}$ and find all indices i such that $S\left[i+1, i+\frac{n}{2}\right]$ has at most k mismatches from $S\left[1, \frac{n}{2}\right]$.

The second process cannot use the same approach, because the k-Mismatch Algorithm reports that index i is a candidate after reading position $\frac{n}{2} + i$, at which point we have already passed $n - i$. This means that the fingerprint of $S[1, n-i]$ cannot be built. For example, see Figure 2.

Thus, for a fixed p in the second process, if we set x to be the largest power of two which does not exceed $n - 2p$, the k-mismatch algorithm could report p. However, we cannot do this without knowing the value of p.

Building off the ideas in [12], we run $\log n$ instances of the algorithm in parallel, with $x = 1, 2, 4, \ldots$, then one of these values of x must correspond to the instance of k-mismatch algorithm that recognizes p and reports it for later verification.

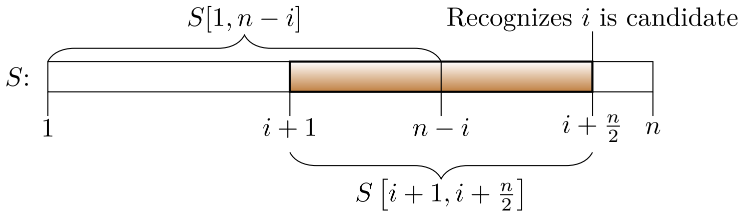

$S[1, n-i]$
Recognizes i is candidate
S:
1
$i+1$
$n-i$
$i+\frac{n}{2}$
n
$S\left[i+1, i+\frac{n}{2}\right]$

Figure 2 When i is recognized as a candidate, the algorithm has already passed $n-i$ and cannot build $S[1, n-i]$.

5.1 Finding small k-periods

We consider all the k-periods p with $p \leq \frac{n}{4}$ for this subsection. Run the k-Mismatch algorithm to find

$$\mathcal{T} = \left\{ i \,\middle|\, i \leq \frac{n}{4}, \mathsf{HAM}\left(S\left[1, \frac{n}{2}\right], S\left[i+1, i+\frac{n}{2}\right]\right) \leq k \right\}.$$

Upon finding an index $i \in \mathcal{T}$, the algorithm uses the fingerprint for $S\left[i+1, i+\frac{n}{2}\right]$ to continue building $S[i+1, n]$. Simultaneously, it builds $S[1, n-i]$, and checks whether $\mathsf{HAM}\left(S[1, n-i], S[i+1, n]\right) \leq k$. The algorithm identifies that $i \in \mathcal{T}$ upon reading character $i + \frac{n}{2} - 1$. Since $i \leq \frac{n}{4}$, then $i + \frac{n}{2} - 1 < \frac{3n}{4} \leq n - i$. Thus, the algorithm can identify i in time to build $S[1, n-i]$. By Theorem 9, these entries can be computed from a sequence of compressed fingerprints.

5.2 Finding large k-periods

Now, consider all the k-periods p with $\frac{n}{4} < p \leq \frac{n}{2}$. Let $I_m = \left[\frac{n}{2} - 2^m + 1, \frac{n}{2} - 2^{m-1}\right]$ and for $1 \leq m \leq \log n - 1$, define

$$\mathcal{T}_m = \left\{ i \,|\, i \in I_m, \mathsf{HAM}\left(S[1, 2^m], S[i+1, i+2^m]\right) \leq k \right\}.$$

Let π_m be a k-period of $S[1, 2^m]$. We first consider the case where $\pi_m \geq \frac{2^m}{4}$ and then the case where $\pi_m < \frac{2^m}{4}$.

▶ **Observation 13** ([7]). *If p is a k-period for $S[1, n/2]$, then each i such that*

$$\mathsf{HAM}\left(S\left[1, \frac{n}{2}\right], S\left[i+1, i+\frac{n}{2}\right]\right) \leq \frac{k}{2}$$

must be at least p symbols apart.

By Observation 13, if $\pi_m \geq \frac{2^m}{4}$, then $|\mathcal{T}_m| \leq 4$. Moreover, we can detect whether $i \in \mathcal{T}_m$ by index $\frac{n}{2} - 2^{m-1} + 2^m$. On the other hand, $n - i \geq \frac{n}{2} + 2^m + 1$, and so we can properly build $S[1, n-i]$.

Now, suppose $\pi_m < \frac{2^m}{4}$. Since $\mathcal{T}_m$ may be linear in size, we use the same trick to obtain a succinct representation, whose properties satisfy those in Section 4, while including a few additional indices. Let $S[2^m + 1, 2^{m+1}] = w_1 w_2 \ldots w_t w'$, where each w_i has length π_m and for $0 \leq d \leq 3k$, let x_d be the largest index such that $S[1, 2^m] \circ w_1 \circ w_2 \circ \cdots \circ w_x$ has d-period π_m.

Let $\mathcal{T}_m = i_1, i_2, \ldots, i_r$ in increasing order. Let $S\left[i_r + 2^m + 1, \frac{n}{2} + 2^m\right] = v_1 v_2 \ldots v_s v'$, where each v_i has length π_m and let y be the largest index such that $S[i_r + 1, i_r + 2^m] \circ v_1 \circ v_2 \circ \cdots \circ v_y$ has $3k$-period π_m.

If $y = s$, then at most k of the substrings v_i can be unique by Observation 2. Moreover, by storing the fingerprints and positions of $\mathcal{O}\left(k^2 \log n\right)$ substrings, as well as v', we can recover the fingerprint of each $S[n-i_{j+1}, n-i_j]$ by Lemma 10. Thus, we keep the fingerprint of $S\left[\frac{n}{2}+1, n-i_r\right]$, and can construct the fingerprint of each $S\left[\frac{n}{2}+1, n-i_j\right]$

On the other hand if $y \neq s$, then for each i_j, let Δ be the number of indices z such that $i_j \leq z \leq i_r$ and $S[z] \neq S[z+\pi_m]$. That is, $\Delta = |\{z | i_j \leq z \leq i_r, S[z] \neq S[z+\pi_m]\}|$. Since π_m is a k-period of $S[1, 2^m]$, $\mathsf{HAM}\left(S[1,2^m], S[i_j+1, i_j+2^m]\right) \leq k$, and each mismatch between $S[1, 2^m]$ and $S[i_j+1, i_j+2^m]$ can cause up to two indices z such that $S[z] \neq S[z+\pi_m]$, then it follows that $0 \leq \Delta \leq 3k$. Then if $y+|r-j| \neq x_{3k-\Delta}$, then $i_j \notin \mathcal{T}_m$, since $x_{3k-\Delta}$ is the largest index with $(3k-\Delta)$-period π_m, while y is the largest index with $3k$-period π_m.

Thus, for each $0 \leq \Delta \leq 2k$, there is at most one index j with $y+|r-j| \neq x_{2k+\Delta}$. Again by Lemma 10, we can compute the fingerprint of $S\left[\frac{n}{2}+1, n-i_j\right]$ by storing the fingerprints and positions of $\mathcal{O}\left(k^2 \log n\right)$ substrings.

Computing each x_d requires determining π_m and the fingerprint of $S[2^m-\pi_m+1, 2^m]$. Since $\pi_m \leq \frac{2^m}{4}$, the algorithm determines π_m by position $\pi_m + 2^m < 2^m - \pi_m + 1$. Thus, the algorithm knows π_m in time to start creating the fingerprint of $S[2^m-\pi_m+1, 2^m]$.

To compute y, we compute the fingerprint of $S[i_r+1, i_r+\pi_m]$. We then compute the fingerprint of each non-overlapping substring of length π_m starting from $i_r+\pi_m$, and compare the fingerprint to the previous fingerprint. We only record the fingerprint of the most recent substring, but keep a running count of the number of mismatches.

▶ **Theorem 14.** *There exists a one-pass algorithm that outputs all the k-periods p of a given string with $p \leq \frac{n}{2}$, and uses $\mathcal{O}\left(k^4 \log^9 n\right)$ bits of space.*

Proof. The process for small k-periods uses $\mathcal{O}\left(k^2 \log^8 n\right)$ bits of space determining $\mathcal{T}$. Verifying whether an index in $\mathcal{T}$ is actually a k-period requires the fingerprints of $\mathcal{O}\left(k^2 \log n\right)$ substrings, each using $\mathcal{O}\left(k \log^6 n\right)$ bits of space (Theorem 5). This adds up to a total of $\mathcal{O}\left(k^3 \log^7 n\right)$ bits of space.

The process for large k-periods has $\log n$ parallel instances of the k-Mismatch algorithm to compute $\mathcal{T}_m$ for $1 \leq m \leq \log n$, using $\mathcal{O}\left(k^2 \log^9 n\right)$ bits of space. To reconstruct the fingerprint of $S[1, n-i]$ for each $i \in \mathcal{T}_m$ the algorithm needs to store the fingerprints of at most $\mathcal{O}\left(k^2 \log n\right)$ unique substrings (Lemma 10). Each fingerprint uses $\mathcal{O}\left(k \log^6 n\right)$ bits of space (Theorem 5) and there can be up to $\mathcal{O}\left(k \log n\right)$ indices in $\mathcal{T}_m$. This adds up to a total of $\mathcal{O}\left(k^4 \log^9 n\right)$ bits of space.

Thus, $\mathcal{O}\left(k^4 \log^9 n\right)$ bits of space suffice for both processes. ◀

6 Lower Bounds

6.1 Lower Bounds for General Periods

Recall the following variant of the Augmented Indexing Problem, denoted $\mathsf{IND}_{n,\delta}$, where Alice is given a string $S \in \Sigma^n$. Bob is given an index $i \in [n]$, as well as $S[1, i-1]$, and must output $S[i]$ correctly with probability at least $1-\delta$.

▶ **Lemma 15** ([24]). *The one-way communication complexity of* $\mathsf{IND}_{n,\delta}$ *is* $\Omega((1-\delta) n \log |\Sigma|)$.

▶ **Theorem 16.** *Any one-pass streaming algorithm which computes the smallest k-period of an input string S requires $\Omega(n)$ space.*

Proof. Consider the following communication game between Alice and Bob, who are given strings A and B respectively. Both A and B have length n, and the goal is to compute the smallest k-period of $a \circ b$. Then we show that any one-way protocol which successfully computes the smallest k-period of $a \circ b$ requires $\Omega(n)$ communication by a reduction from the augmented indexing problem.

Suppose Alice gets a string $S \in \{0,1\}^n$, while Bob gets an index $i \in [n-1]$ and $S[1, i-1]$. Let $\mathbf{u}$ be the binary negation of $S[1]$, i.e., $\mathbf{u} = 1 - S[1]$. Then Alice sets $A = (S[1])^k (S[2])^k \ldots (S[n])^k$ and Bob sets $B = \mathbf{u}^{k(n-i)} \circ (S[1])^k (S[2])^k \ldots (S[i-1])^k \circ \mathbf{1}^k$ so that both A and B have length kn. Moreover, the smallest k-period of $A \circ B$ is $k(2n-i)$ if and only if $S[i] = 1$. ◀

6.2 Lower Bounds for Small Periods

We now show that for $k = o(\sqrt{n})$, even given the promise that the smallest k-period is at most $\frac{n}{2}$, any randomized algorithm which computes the smallest k-period with probability at least $1 - \frac{1}{n}$ requires $\Omega(k \log n)$ space. By Yao's Minimax Principle [28], it suffices to show a distribution over inputs such that every deterministic algorithm using less than $\frac{k \log n}{6}$ bits of memory fails with probability at least $\frac{1}{n}$.

Define an infinite string $1^1 0^1 1^2 0^2 1^3 0^3 \ldots$, as in [16], and let ν be the prefix of length $\frac{n}{4}$. Let X be the set of binary strings of length $\frac{n}{4}$ at Hamming distance $\frac{k}{2}$ from ν. Given $x \in X$, let Y_x be the set of binary strings of length $\frac{n}{4}$ with either $\mathsf{HAM}(x,y) = \frac{k}{2}$ or $\mathsf{HAM}(x,y) = \frac{k}{2} + 1$. We pick (x,y) uniformly at random from (X, Y_x).

▶ **Theorem 17.** *Given an input $x \circ y$, any deterministic algorithm $\mathcal{D}$ that uses less than $\frac{k \log n}{6}$ bits of memory cannot correctly output whether $\mathsf{HAM}(x,y) = \frac{k}{2}$ or $\mathsf{HAM}(x,y) > \frac{k}{2}$ with probability at least $1 - \frac{1}{n}$, for $k = o(\sqrt{n})$.*

Proof. Note that $|X| = \binom{n/4}{k/2}$. By Stirling's approximation, $|X| \geq \left(\frac{n}{2k}\right)^{k/2} \geq \left(\frac{n}{4}\right)^{k/4}$ for $k = o(\sqrt{n})$.

Because $\mathcal{D}$ uses less than $\frac{k \log n}{6}$ bits of memory, then $\mathcal{D}$ has at most $2^{\frac{k \log n}{6}} = n^{k/6}$ unique memory configurations. Since $|X| \geq \left(\frac{n}{4}\right)^{k/4}$, then there are at least $\frac{1}{2}(|X| - n^{k/6}) \geq \frac{|X|}{4}$ pairs x, x' such that $\mathcal{D}$ has the same configuration after reading x and x'. We show that $\mathcal{D}$ errs on a significant fraction of these pairs x, x'.

Let $\mathcal{I}$ be the positions where either x or x' differ from ν, so that $\frac{k}{2} + 1 \leq |\mathcal{I}| \leq k$. Observe that if $\mathsf{HAM}(x,y) = \frac{k}{2}$, but x and y do not differ in any positions of $\mathcal{I}$, then $\mathsf{HAM}(x',y) > \frac{k}{2}$. Recall that $\mathcal{D}$ has the same configuration after reading x and x', so then $\mathcal{D}$ has the same configuration after reading $x \circ y$ and $x' \circ y$. But since $\mathsf{HAM}(x,y) = \frac{k}{2}$ and $\mathsf{HAM}(x',y) > \frac{k}{2}$, then the output of $\mathcal{D}$ is incorrect for either $x \circ y$ or $x' \circ y$.

For each pair (x, x'), there are $\binom{n/4 - |\mathcal{I}|}{k/2} \geq \binom{n/4-k}{k/2}$ such y with $\mathsf{HAM}(x,y) = \frac{k}{2}$, but x and y do not differ in any positions of $\mathcal{I}$. Hence, there are $\frac{|X|}{4}\binom{n/4-k}{k/2}$ strings $S(x,y)$ for which $\mathcal{D}$ errs. Recall that y satisfies either $\mathsf{HAM}(x,y) = \frac{k}{2}$ or $\mathsf{HAM}(x,y) = \frac{k}{2} + 1$ so that there are $|X|\left(\binom{n/4}{k/2} + \binom{n/4}{k/2+1}\right)$ strings $x \circ y$ in total. Thus, the probability of error is at least

$$\begin{aligned}
\frac{\frac{|X|}{4}\binom{n/4-k}{k/2}}{|X|\left(\binom{n/4}{k/2} + \binom{n/4}{k/2+1}\right)} &= \frac{1}{4} \cdot \frac{\binom{n/4-k}{k/2}}{\binom{n/4+1}{k/2+1}} = \frac{(k/2+1)}{4} \frac{(n/4 - 3k/2 + 1) \ldots (n/4 - k)}{(n/4 - k/2 + 1) \ldots (n/4 + 1)} \\
&\geq \frac{k/2+1}{n+4} \left(\frac{n/4 - 3k/2 + 1}{n/4 - k/2 + 1}\right)^{k/2} = \frac{k+2}{2n+8}\left(1 - \frac{k}{n/4 - k/2 + 1}\right)^{k/2} \\
&\geq \frac{k+2}{2n+8}\left(1 - \frac{k^2}{n/2 - k + 2}\right) \geq \frac{1}{n}
\end{aligned}$$

where the last line holds for large n, from Bernoulli's Inequality and $k = o(\sqrt{n})$. ◀

▶ **Lemma 18.** *For $k = o(\sqrt{n})$, any k-period of the string $S(x, y) = x \circ y \circ x \circ x$ is at least $\frac{n}{4}$.*

Proof. We show that stronger result that if $p < \frac{n}{4}$, $k > 2$, and $n > 4(18k+1)(18k+2)$, then $|\{z|S[z] \neq S[z+p]\}| > \sqrt{\frac{n}{8}} > k$, for $k = o(\sqrt{n})$.

Let $T = \nu \circ \nu \circ x \circ x$ and for each z, consider $T[z]$ and $T[z+p]$. For each $j > 0$, some position $z + p$ in $1^{2j}0^{2j}1^{2j+1}0^{2j+1}$ in the second ν corresponds with a mismatch in z. Since $\mathsf{HAM}(x, \nu) = \frac{k}{2}$ and $\mathsf{HAM}(x, y) \leq \frac{k}{2} + 1$, then $\mathsf{HAM}\left(S\left[1, \frac{n}{2}\right], T\left[1, \frac{n}{2}\right]\right) \leq \frac{3k}{2} + 1$. Each mismatch between S and T can cause at most two indices z for which $T[z] \neq T[z+p]$ but $S[z] = S[z+p]$. Thus, by setting $j = 6k > 2\left(\frac{3k}{2} + 1\right) + 2k$, we have that for $\frac{n}{4} > (12k+1)(12k+2)$, there are at least $6k$ indices z for which $T[z] \neq T[z+p]$, and thus at least $2k$ indices for which $S[z] \neq S[z+p]$. ◀

▶ **Corollary 19.** *If $\mathsf{HAM}(x, y) = \frac{k}{2}$, then the string $S(x, y) = x \circ y \circ x \circ x$ has period $\frac{n}{4}$. On the other hand, if $\mathsf{HAM}(x, y) = \frac{k}{2} + 1$, then $S(x, y)$ has period greater than $\frac{n}{4}$.*

▶ **Theorem 20.** *For $k = o(\sqrt{n})$ with $k > 2$, any one-pass streaming algorithm which computes the smallest k-period of an input string S with probability at least $1 - \frac{1}{n}$ requires $\Omega(k \log n)$ space, even under the promise that the k-period is at most $\frac{n}{2}$.*

Proof. By Theorem 17, any algorithm using less than $\frac{k \log n}{6}$ bits of memory cannot distinguish between $\mathsf{HAM}(x, y) = \frac{k}{2}$ and $\mathsf{HAM}(x, y) = \frac{k}{2} + 1$ with probability at least $1 - 1/n$. Thus, no algorithm can distinguish whether the period of $S(x, y)$ is $\frac{n}{4}$ with probability at least $1 - 1/n$ while using less than $\frac{k \log n}{6}$ bits of memory. ◀

References

1 Amihood Amir, Estrella Eisenberg, and Avivit Levy. Approximate periodicity. *Algorithms and Computation*, pages 25–36, 2010.

2 Alexandr Andoni, Assaf Goldberger, Andrew McGregor, and Ely Porat. Homomorphic fingerprints under misalignments: sketching edit and shift distances. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 931–940, 2013.

3 Alberto Apostolico and Zvi Galil, editors. *Pattern Matching Algorithms.* Oxford University Press, Oxford, UK, 1997.

4 Petra Berenbrink, Funda Ergün, Frederik Mallmann-Trenn, and Erfan Sadeqi Azer. Palindrome recognition in the streaming model. In *31st International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 149–161, 2014.

5 Raphaël Clifford, Klim Efremenko, Ely Porat, and Amir Rothschild. From coding theory to efficient pattern matching. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 778–784, 2009.

6 Raphaël Clifford, Allyx Fontaine, Ely Porat, Benjamin Sach, and Tatiana A. Starikovskaya. Dictionary matching in a stream. In *Algorithms – ESA 23rd Annual European Symposium, Proceedings*, pages 361–372, 2015.

7 Raphaël Clifford, Allyx Fontaine, Ely Porat, Benjamin Sach, and Tatiana A. Starikovskaya. The k-mismatch problem revisited. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA*, pages 2039–2052, 2016.

8 Raphaël Clifford, Markus Jalsenius, Ely Porat, and Benjamin Sach. Space lower bounds for online pattern matching. *Theoretical Computer Science*, 483:68–74, 2013.

9 Michael S. Crouch and Andrew McGregor. Periodicity and cyclic shifts via linear sketches. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques – 14th International Workshop, APPROX, and 15th International Workshop, RANDOM. Proceedings*, pages 158–170, 2011.

10 Mohamed G. Elfeky, Walid G. Aref, and Ahmed K. Elmagarmid. STAGGER: periodicity mining of data streams using expanding sliding windows. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM)*, pages 188–199, 2006.

11 Funda Ergün, Elena Grigorescu, Erfan Sadeqi Azer, and Samson Zhou. Streaming periodicity with mismatches, 2017. URL: `http://homes.soic.indiana.edu/fergun/PUBLICATIONS/mismatchperiodicity.pdf`.

12 Funda Ergün, Hossein Jowhari, and Mert Saglam. Periodicity in streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 13th International Workshop, APPROX 2010, and 14th International Workshop, RANDOM 2010. Proceedings*, pages 545–559, 2010.

13 Funda Ergün, S. Muthukrishnan, and Süleyman Cenk Sahinalp. Periodicity testing with sublinear samples and space. *ACM Trans. Algorithms*, 6(2):43:1–43:14, 2010.

14 Zvi Galil and Joel Seiferas. Time-space-optimal string matching. *Journal of Computer and System Sciences*, 26(3):280–294, 1983.

15 Pawel Gawrychowski. Optimal pattern matching in LZW compressed strings. *ACM Transactions on Algorithms (TALG)*, 9(3):25, 2013.

16 Pawel Gawrychowski, Oleg Merkurev, Arseny M. Shur, and Przemyslaw Uznanski. Tight tradeoffs for real-time approximation of longest palindromes in streams. In *27th Annual Symposium on Combinatorial Pattern Matching, CPM*, pages 18:1–18:13, 2016.

17 Shay Golan, Tsvi Kopelowitz, and Ely Porat. Streaming Pattern Matching with d Wildcards. In *24th Annual European Symposium on Algorithms (ESA)*, pages 44:1–44:16, 2016.

18 Elena Grigorescu, Erfan Sadeqi Azer, and Samson Zhou. Streaming for aibohphobes: Longest palindrome with mismatches. *CoRR*, abs/1705.01887, 2017. URL: `http://arxiv.org/abs/1705.01887`.

19 Piotr Indyk, Nick Koudas, and S. Muthukrishnan. Identifying representative trends in massive time series data sets using sketches. In *VLDB, Proceedings of 26th International Conference on Very Large Data Bases*, pages 363–372, 2000.

20 Richard M. Karp and Michael O. Rabin. Efficient randomized pattern-matching algorithms. *IBM Journal of Research and Development*, 31(2):249–260, 1987.

21 Donald E. Knuth, James H. Morris Jr., and Vaughan R. Pratt. Fast pattern matching in strings. *SIAM J. Comput.*, 6(2):323–350, 1977.

22 Donald E. Knuth, James H. Morris Jr., and Vaughan R. Pratt. Fast pattern matching in strings. *SIAM journal on computing*, 6(2):323–350, 1977.

23 Oded Lachish and Ilan Newman. Testing periodicity. *Algorithmica*, 60(2):401–420, 2011.

24 Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing*, pages 103–111, 1995.

25 Benny Porat and Ely Porat. Exact and approximate pattern matching in the streaming model. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS*, pages 315–323, 2009.

26 Ely Porat and Ohad Lipsky. Improved sketching of hamming distance with error correcting. In *Annual Symposium on Combinatorial Pattern Matching*, pages 173–182, 2007.

27 Jakub Radoszewski and Tatiana Starikovskaya. Streaming k-mismatch with data recovery and applications. *arXiv preprint arXiv:1607.05626*, 2016.

28 Andrew Chi-Chih Yao. Probabilistic computations: Toward a unified measure of complexity (extended abstract). In *18th Annual Symposium on Foundations of Computer Science, FOCS*, pages 222–227, 1977.

A Structural Properties of k-Periodic Strings

In this section, we show several steps towards proving Theorem 9. We defer the detailed proofs to the full version [11].

We first show Theorem 23, which assumes there are only two candidate k-periods and both are small. We then relax these conditions and prove Theorem 30, which does not restrict the number of candidate k-periods, but still assumes that their magnitudes are small. Theorem 9 considers all candidate k-periods in some interval. We use the fact that the *difference* between these candidates is small, thus meeting the conditions of Theorem 30, although with an increase in the number of mismatches.

To show that the greatest common divisor d of any two reasonably small candidates $p < q$ for k-periods is also a $(16k^2 + 1)$-period (Theorem 23), we consider the cases where either all candidates are less than $(2k + 1)d$ (Lemma 24) or some candidate is at least $(2k + 1)d$ (Lemma 25).

In the first case, where all candidate period are less than $(2k + 1)d$, we partition the string into disjoint intervals of a certain length, followed by partitioning the intervals further into congruence classes. We show in Lemma 22 that any partition which contains an index i such that $S[i] \neq S[i+d]$ must also contain an index j which is a mismatch from some symbol p or q distance away. Since there are at most $2k$ indices j, we can then bound the number of such partitions, and then extract an upper bound on the number of such indices i.

In the second case, where some candidate is at least $(2k + 1)d$, our argument relies on forming a grid (such as in Figure 3) where adjacent points are indices which either differ by p or q. We include $2k + 1$ rows and columns in this grid. Since $\frac{q}{d} \geq 2k + 1$, then no index in S is represented by multiple points in the grid. We call an edge between adjacent points "bad" if the two corresponding indices form a mismatch.

▶ **Observation 21.** *$S[i] \neq S[i + d]$ only if each path between i and $i + d$ contains a bad edge.*

Our grid contains at most $2k$ bad edges, since p and q are both k-periods, and each index is represented at most once. We then show that for all but at most $(16k^2 + 1)$ indices i, there exists a path between indices i and $i + d$ that avoids bad edges. Therefore, there are at most $(16k^2 + 1)$ indices i such that $S[i] \neq S[i + d]$, which shows that d is an $(16k^2 + 1)$-period.

Before proving Lemma 24, we first show a number theoretic result that given integers i, p, q, we can repeatedly hop by distance p or q, starting from i, ending at $i + \mathsf{gcd}\,(p, q)$, all the while staying in a "small" interval.

▶ **Lemma 22.** *Suppose $p < q$ are two positive integers with $\mathsf{gcd}\,(p, q) = d$. Let i be an integer such that $1 \leq i \leq p + q - d$. Then there exists a sequence of integers $i = t_0, \dots, t_m = i + d$ where $|t_i - t_{i+1}|$ is either p or q, and $1 \leq t_i < p + q$. Furthermore, each integer is congruent to $i \pmod d$. In other words, any interval of length $p + q$ which contains indices $i, i + d$ such that $S[i] \neq S[i + d]$ also contains an index j such that either $S[j] \neq S[j + p]$ or $S[j] \neq S[j + q]$.*

Proof. Since d is the greatest common divisor of p and q, then there exist integers a, b such that $ap + bq = d$. Suppose $a > 0$. Then consider the sequence $t_i = t_{i-1} + p$ if $1 \leq t_{i-1} \leq q$. Otherwise, if $t_{i-1} > q$, let $t_i = t_{i-1} - q$. Then clearly, each $|t_i - t_{i+1}|$ is either p or q, and $1 \leq t_i < p + q$. That is, each t_i either increases the coefficient of p by one, or decreases the coefficient of q by one. Thus, at the last time the coefficient of p is a, $t_i = ap + bq = d$, since any other coefficient of q would cause either $t_i > q$ or $t_i < 1$. Hence, terminating the sequence at this step produces the desired output, and a similar argument follows if $b > 0$ instead of $a > 0$. Since $p \equiv q \equiv 0 \pmod d$, then all integers in these sequence are congruent to $i \pmod d$. ◀

We now prove that the greatest common divisor d of any two reasonably small candidates p, q for k-periods is also a $(16k^2+1)$-period.

▶ **Theorem 23.** *For any* $1 \le x \le \frac{n}{2}$*, let* $\mathcal{I} = \left\{ i \,\middle|\, i \le \frac{x}{4k+2}, \mathsf{HAM}\left(S[1,x], S[i+1, i+x]\right) \le k \right\}$*. For any two* $p, q \in \mathcal{I}$ *with* $p < q$*, their greatest common divisor,* $d = \mathsf{gcd}\,(p,q)$ *satisfies*

$$\mathsf{HAM}\left(S[1,x], S[d+1, d+x]\right) \le (16k^2+1).$$

We now proceed to the proof of Theorem 23 for the case $q < (2k+1)d$.

▶ **Lemma 24.** *Theorem 23 holds when* $q < (2k+1)d$*.*

Proof. If $x \le 16k^2$, then clearly there are at most $16k^2$ indices i such that $S[i] \ne S[i+d]$, and so d is a $(16k^2+1)$-period. Otherwise, suppose $x > 16k^2+1$, and by way of contradiction, that there are at least $16k^2+1$ indices i such that $S[i] \ne S[i+d]$.

Consider the following two classes of intervals of length $\frac{p+q}{2}$:

$$\mathcal{I}_1 = \left[1, \frac{p+q}{2}\right], \left[p+q+1, \frac{3(p+q)}{2}\right], \left[2(p+q)+1, \frac{5(p+q)}{2}\right], \ldots$$

and

$$\mathcal{I}_2 = \left[\frac{p+q}{2}+1, p+q\right], \left[\frac{3(p+q)}{2}+1, 2(p+q)\right], \left[\frac{5(p+q)}{2}+1, 3(p+q)\right], \ldots.$$

If there are at least $16k^2+1$ indices i such that $S[i] \ne S[i+d]$, then either $\mathcal{I}_1$ or $\mathcal{I}_2$ contains at least $8k^2+1$ of these indices.

Suppose $\mathcal{I}_1$ has at least $8k^2+1$ indices i such that $S[i] \ne S[i+d]$. Now, consider the disjoint intervals of length $p+q$: $[1, p+q]$, $[p+q+1, 2(p+q)]$, $[2(p+q)+1, 3(p+q)]$, …. Furthermore, for each of these intervals, consider the congruence classes modulo d. Since $x > 16k^2+1$ and each of these congruence classes within an intervals have $\frac{p+q}{d} < \frac{2q}{d} \le 2(2k) = 4k$ indices, then $S[1,x]$ certainly contains at least $2k+1$ of these congruence classes.

If $\mathcal{I}_1$ has at least $8k^2+1$ indices i such that $S[i] \ne S[i+d]$ and each congruence class within an interval contains less than $4k$ indices, then there are at least $2k+1$ congruence classes containing such an index i. Because each of these indices occur within $\mathcal{I}_1$, it follows that both i and $i+d$ are contained within the interval (and therefore, the same congruence class). By Lemma 22, each congruence class within an interval containing indices i and $i+d$ $S[i] \ne S[i+d]$ also contains an index j such that either $S[j] \ne S[j+p]$ or $S[j] \ne S[j+q]$. Since there are at least $2k+1$ congruence classes within intervals, then there are at least $2k+1$ such indices j. This either contradicts that there are at most k indices j such that $S[j] \ne S[j+p]$ or there are at most k indices j such that $S[j] \ne S[j+q]$.

The proof for the case where $\mathcal{I}_2$ has at least $8k^2+1$ indices i such that $S[i] \ne S[i+d]$ is symmetric. ◀

The following lemma considers the case where at least one of candidate periods p or q is at least $(2k+1)d$. Without loss of generality, assume $q \ge (2k+1)d$. We form a grid, such as in Figure 3, where adjacent points in the grid correspond to indices which either differ by p or q. An edge between adjacent points is "bad" if the two corresponding indices form a mismatch.

From Observation 21, $S[i] \ne S[i+d]$ only if each path between i and $i+d$ contains a bad edge. Thus, if $S[i] \ne S[i+d]$, then the point in the grid corresponding to i must be contained in some region whose boundary is formed by bad edges. We partition the indices into congruence classes modulo d, count the number of mismatches in each class, and aggregate the results.

That is, in a particular congruence class, we assume p is a k_1-period, and q is a k_2-period, where $k_1, k_2 \leq k$. Then the grid contains at most $k_1 + k_2$ bad edges, which bounds the perimeter of the regions. From this, we deduce a generous bound of $(16k_1k_2 + 1)$ on the number of points inside these regions, which is equivalent to the number of indices i such that $S[i] \neq S[i + d]$ in the congruence class. We then aggregate over all congruence classes to show that d is a $(16k^2 + 1)$-period.

▶ **Lemma 25.** *Let $p \leq q$ and k be positive integers with $q \geq (2k + 1)d$ and let $d = \mathsf{gcd}\,(p, q)$. Given a string S and an integer $0 \leq m < d$, let there be $k_1 > 0$ indices $i \equiv m \pmod d$ such that $S[i] \neq S[i + p]$ and $k_2 > 0$ indices $i \equiv m \pmod d$, not necessarily disjoint, such that $S[i] \neq S[i + q]$ and $k_1, k_2 \leq k$. If $d = \mathsf{gcd}\,(p, q)$, then there exist at most $8k_1k_2 + 1$ indices $i \equiv m \pmod d$ such that $S[i] \neq S[i + d]$.*

Proof. Consider a pair of indices $(i, i + d)$ with $S[i] \neq S[i + d]$ in congruence class $m \pmod d$. We ultimately want to build a grid of "large" size around i, but this may result in illegal indices if i is too small or too large. Therefore, we first consider the case where $k(p + q) \leq i \leq x - k(p + q)$, where we can place i in the center of the grid. We then describe a similar argument with modifications for $i < k(p + q)$ or $i > x - k(p + q)$, when we must place i near the periphery of the grid.

Given index i with $k(p + q) \leq i \leq x - k(p + q)$, we define a grid on a subset of indices of $S[1, x]$. The node at the center is i and for any node j, the nodes $j + p$, $j + q$, $j - p$ and $j - q$ are the top, right, bottom and left neighbors of j, respectively. See Figure 3 for example of such a grid.

We include $(2k + 1)$ rows and columns in this grid, where i is the intersection of the middle row and the middle column. Note that since $k(p + q) \leq i \leq x - k(p + q)$, all points in the grid correspond to indices of S.

▶ **Claim 26.** *No indices of S correspond to multiple points in the grid.*

Proof. Suppose, by way of contradiction, there exists some index j which is represented by multiple points in the grid. That is, $j = i + a_1p + b_1q = i + a_2p + b_2q$ with $a_1 \neq a_2$. Since $d = \mathsf{gcd}\,(p, q)$, there exist integers r, s with $p = rd$, $q = sd$, and $\mathsf{gcd}\,(r, s) = 1$. Then $(a_1 - a_2)p = (b_2 - b_1)q$ so $(a_1 - a_2)r = (b_2 - b_1)s$. Because $\mathsf{gcd}\,(r, s) = 1$, it follows that $(a_1 - a_2)$ is divisible by $s = \frac{q}{d} \geq 2k + 1$. Therefore, $|a_1 - a_2| \geq 2k + 1$, and so a_1 and a_2 are at least $2k + 1$ columns apart. However, this contradicts both points being in the grid, since the grid contains exactly $2k + 1$ columns. ◀

▶ **Claim 27.** *There exist at least $k + 1$ rows and $k + 1$ columns in the grid that do not contain any bad edge.*

Proof. Since $\mathsf{HAM}\,(S[1, x], S[\alpha + 1, \alpha + x]) \leq k$, for $\alpha = p, q$, there are at most k indices i for which $S[i] \neq S[i + p]$ or $S[i] \neq S[i + q]$. By Claim 26, each index is represented at most once. Hence, there are at most k vertical bad edges and at most k horizontal bad edges in this grid. Because the grid contains $2k + 1$ rows and columns, then there exist at least $k + 1$ rows and columns in the grid that do not contain any bad edge. ◀

We call these rows and columns *no-change*.

▶ **Claim 28.** *If there exists a path between i and a no-change row or column in a grid containing i avoiding bad edges, and a path between $i+d$ and a no-change row or column in a grid containing $i+d$ avoiding bad edges, then there exists a path between i and $i+d$ avoiding bad edges.*

Proof. Notice that some no-change row in the grid centered at i must also be a no-change row in the grid centered at $i+q$, since there are at least $k+1$ no-change rows in each grid, but the two grids overlap in $2k+1$ rows. Similarly, some no-change column in the grid centered at i must also be a no-change row in the grid centered at $i+p$. These common no-change rows and columns allow traversal between grids, as we can freely traverse between any no-change rows and columns while avoiding bad edges. Thus, if we can traverse from i to any no-change row in the first grid, we can ultimately reach any no-change row in the final grid containing $i+d$ while avoiding all bad edges. Finally, if we can traverse between $i+d$ and any no-change row in the final grid, then there exists a path between i and $i+d$ without any bad edges. ◀

This construction describes a possible path from i to $i+d$ with the help of these no-change rows and columns between grids. Notice that it is possible that there is no path from i to $i+d$ simply because a lot of bad edges have surrounded node i or $i+d$. (This is a necessary but not sufficient condition.)

We use the term *isolated* node, to describe any node which is in a region enclosed by bad edges. Note that points in such enclosed regions are also possibly part of mismatched indices $(j, j+d)$. We argue that the most number of unique indices which can enclosed with k_1 vertical edges and k_2 horizontal edges is $\frac{k_1k_2}{2}+2k_1+2k_2$, even on an extended grid with no boundaries and multiple vertices/edges which correspond to the same index.

▶ **Claim 29.** *The number of isolated nodes is at most $\frac{k_1k_2}{2}+2k_1+2k_2$.*

We sketch the details of the proof of Claim 29, with full details provided in [11]. The total area of regions enclosed by at most k_1 vertical bad edges and at most k_2 horizontal bad edges is at most $\frac{k_1k_2}{4}$. Thus, the number of isolated nodes cannot exceed $\frac{k_1k_2}{4}$.

The number of $(i, i+d)$ mismatches is at most double the number of isolated nodes (if i is isolated, both $(i, i+d)$ and $(i-d, i)$ may be mismatches) plus the number of mismatched edges. The former is bounded by $\frac{k_1k_2}{4}$, the latter by k_1+k_2. See Figure 3 for example.

We defer the casework for $i<k(p+q)$ and $i>x-k(p+q)$ to the full version [11]. ◀

The proof of Theorem 23 follows by aggregating each congruence class with mismatched indices, handled in Lemma 25.

We generalize Theorem 23 by showing that the greatest common divisor of any $m\geq 2$ reasonably small candidates for k-periods is also a $(2mk^2+1)$-period. We emphasize that it is sufficient for $m\leq\log n$, since the greatest common divisor can change at most $\log n$ times.

▶ **Theorem 30.** *Let $\mathcal{I}=\left\{i\,\middle|\,i\leq\frac{x}{2(mk+1)},\mathsf{HAM}\left(S[1,x],S[i+1,i+x]\right)\leq k\right\}$. The greatest common divisor of any $p_1,\ldots,p_m\in\mathcal{I}$, $d=\mathsf{gcd}\left(p_1,\ldots,p_m\right)$, satisfies*

$$\mathsf{HAM}\left(S[1,x],S[d+1,d+x]\right)\leq 8mk^2+1.$$

Although the pairwise greatest common divisor between two candidates p_i and p_j is no longer d, considering $\delta=\mathsf{gcd}\left(p_1,p_m\right)$ suffices for the analysis. If $\frac{p_m}{\delta}<2k+1$, then the proof is similar to that of Lemma 24. Otherwise if $\frac{p_m}{\delta}\geq 2k+1$, the proof is similar to that of Lemma 25. We show a k^2 bound on the volume of an enclosed region, whose surface area

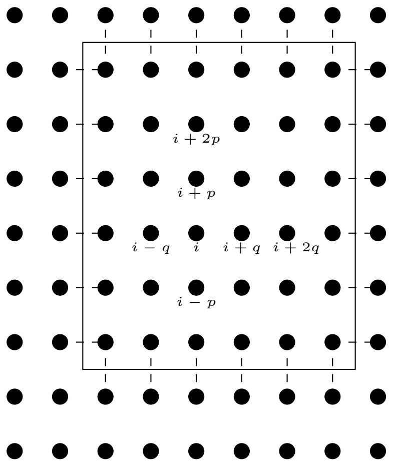


Figure 3 The dashed lines are bad edges. The total area of the enclosed regions can be at most k^2 if the perimeter is at most $4k$.

is at most mk, within a hypergrid. This yields a related bound on the number of isolated nodes.

Observe that Theorem 9 relaxes the constraints of Theorem 30. The full details for the proof of Theorem 23, Theorem 30, and Theorem 9 are provided in [11].

Locality via Partially Lifted Codes*

S. Luna Frank-Fischer[1], Venkatesan Guruswami[†2], and Mary Wootters[‡3]

1 **Computer Science Department, Stanford University, Stanford, CA, USA**
`luna16@stanford.edu`

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

3 **Computer Science Department, Stanford University, Stanford, CA, USA**
`marykw@stanford.edu`

Abstract

In error-correcting codes, *locality* refers to several different ways of quantifying how easily a small amount of information can be recovered from encoded data. In this work, we study a notion of locality called the s-Disjoint-Repair-Group Property (s-DRGP). This notion can interpolate between two very different settings in coding theory: that of Locally Correctable Codes (LCCs) when s is large – a very strong guarantee – and Locally Recoverable Codes (LRCs) when s is small – a relatively weaker guarantee. This motivates the study of the s-DRGP for intermediate s, which is the focus of our paper. We construct codes in this parameter regime which have a higher rate than previously known codes. Our construction is based on a novel variant of the *lifted codes* of Guo, Kopparty and Sudan. Beyond the results on the s-DRGP, we hope that our construction is of independent interest, and will find uses elsewhere.

1998 ACM Subject Classification E.4 Error Control Codes

Keywords and phrases Error correcting codes, locality, lifted codes

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.43

1 Introduction

In the theory of error correcting codes, *locality* refers to several different ways of quantifying how easily a small amount of information can be recovered from encoded data. Slightly more formally, suppose that $\mathcal{C} \subset \Sigma^N$ is a *code* over an alphabet Σ; that is, $\mathcal{C}$ is any subset of Σ^N. Suppose that $c \in \mathcal{C}$, and that we have query access to a noisy version $\tilde{c}$ of c. We are tasked with finding $c_i \in \Sigma$ for some $i \in [N]$. Informally, we say that the code $\mathcal{C}$ exhibits good *locality* if we may recover c_i using very few queries to $\tilde{c}$. Of course, the formal definition of locality in this set-up depends on the nature of the noise, and the question is interesting for a wide variety of noise models.

One (extremely strong) model of noise is that handled by *Locally Correctable Codes* (LCCs), which have been extensively studied in theoretical computer science for over 15 years. This model is motivated by a variety of applications in theoretical computer science and cryptography, including probabilistically checkable proofs (PCPs), derandomization, and private information retrieval (PIR); we refer the reader to [30] for an excellent survey

[*] Full version available at `https://arxiv.org/abs/1704.08627`.

[†] Research supported in part by NSF grants CCF-1563742 and CCF-1422045.

[‡] Research supported in part by NSF grants DMS-1400558 and CCF-1657049.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 43; pp. 43:1–43:17
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

on LCCs. In the LCC setting, $\tilde{c} \in \Sigma^N$ has a constant fraction of errors: that is, we are guaranteed that the Hamming distance between $\tilde{c}$ and c is no more than δN, for some small constant $\delta > 0$. The goal is to recover c_i with high probability from $Q = o(N)$ randomized queries to $\tilde{c}$.

Another (much weaker) model of noise is that handled by *Locally Recoverable Codes* (LRCs) and related notions, which have been increasingly studied recently motivated by applications in distributed storage [14, 10, 23]. In this model, $\tilde{c} \in (\Sigma \cup \{\perp\})^N$ has a constant *number* of *erasures*: that is, we are guaranteed that the number of $\perp$ symbols in $\tilde{c}$ is at most some constant $e = O(1)$, and further that $c_i = \tilde{c}_i$ whenever $\tilde{c}_i \neq \perp$. As before, the goal is to recover c_i using as few queries as possible to $\tilde{c}$. Batch codes [15, 7] and PIR codes [8, 6] are other variants that are interesting in this parameter regime.

A key question in both of these lines of work is how to achieve these recovery guarantees with as high a *rate* as possible. The rate of a code $\mathcal{C} \in \Sigma^N$ is defined to be the ratio $\log_{|\Sigma|}(|\mathcal{C}|)/N$; it captures how much information can be transmitted using such a code. In other words, given N, we seek to find a $\mathcal{C} \subseteq \Sigma^N$ with good locality properties, so that $|\mathcal{C}|$ is as large as possible.

In the context of the second line of work above, recent work [28, 21, 25, 27, 1, 8] has studied (both implicitly and explicitly) the trade-off between rate and something called the s-Disjoint-Repair-Group-Property (s-DRGP) for small s. Informally, $\mathcal{C}$ has the s-DRGP if any symbol c_i can be obtained from s disjoint query sets $c|_{S_1}, c|_{S_2}, \ldots, c|_{S_s}$ for $S_i \subseteq [N]$. (Notice that there is no explicit bound on the size of these query sets, just that they must be disjoint).

One observation which we will make below is that the s-DRGP provides a natural way to interpolate between the first (LCC) setting and the second (LRC) setting above. More precisely, while the LRC setting corresponds to small s (usually, $s = O(1)$), the LCC setting is in fact equivalent to the case when $s = \Omega(N)$. This observation motivates the study of *intermediate* s, which is the goal in this paper.

Contributions

Before we give a more detailed overview of previous work, we outline the main contributions of this paper.

1. **Constructions of codes with the s-DRGP for intermediate s.** We give a construction of a family of codes which have the s-DRGP for $s \sim N^{1/4}$. Our construction can achieve a higher rate than previous constructions with the same property.
2. **A general framework, based on partially lifted codes.** Our codes are based on a novel variant of the *lifted codes* of Guo, Kopparty and Sudan [12]. In that work, with the goal of obtaining LCCs, the authors showed how to construct affine-invariant codes by a “lifting" operation. In a bit more detail, their codes are multivariate polynomial codes, whose entries are indexed by $\mathbb{F}_q^m$ (so $N = q^m$). These codes have the property that, the restriction of each codeword to every line in $\mathbb{F}_q^m$ is a codeword of a suitable univariate polynomial code. (For example, a Reed-Muller code is a *subset* of a lift of a Reed-Solomon code; the beautiful insight of [12] is that in fact the lifted code may be much larger.)
   In our work, we introduce a version of the lifting operation where we only require that the restriction to *some* lines lie in the smaller code, rather than the restriction to *all* lines; we call such codes “partially lifted codes." This partial lifting operation potentially allows for higher-rate codes, and, as we will see, it naturally gives rise to codes with the s-DRGP. One of our main contributions is the introduction of these codes, as well as some machinery which allows us to control their rate. We instantiate this machinery with a particular

example, in order to obtain the construction advertised above. We can also recover previous results in the context of this machinery.

3. **Putting the study of the s-DRGP in the context of LRCs and LCCs.** While the s-DRGP has been studied before, to the best of our knowledge, it is not widely viewed as a way to interpolate between the two settings described above. One of the goals of this paper is to highlight this property and its potential importance to our understanding of locality, both from the LRC/batch code/PIR code side of things, and from the LCC side.

1.1 Background and related work

As mentioned above, in this work we study the s-Disjoint-Repair-Group Property (s-DRGP). We begin our discussion of the s-DRGP with some motivation from the LRC end of the spectrum, from applications in distributed storage. The following model is common in distributed storage: imagine that each server or node in a distributed storage system is holding a single symbol of a codeword $c \in \mathcal{C}$. Over time, nodes fail, usually one at a time, and we wish to repair them (formally, recovering c_i for some i). Moreover, when they fail, it is clear that they have failed. This naturally gives rise to the second parameter regime described above, where $\tilde{c}$ has a constant number of erasures.

Locally recoverable (or repairable) codes (LRCs) [14, 10, 23] were introduced to deal with this setting. The guarantee of an LRC[1] with locality Q is that for any $i \in \{1, \ldots, n\}$, the i'th symbol of the codeword can be determined from a set of at most Q other symbols. There has been a great deal of work recently aimed at pinning down the trade-offs between rate, distance, and the locality parameter Q in LRCs. At this point, we have constructions which have optimal trade-offs between these parameters, as well as reasonably small alphabet sizes [26]. However, there are still many open questions; a major question is how to handle a small number of erasures, rather than a single erasure. This may result from either multiple node failures, or from “hot" data being overloaded with requests. There are several approaches in the literature, but the approach relevant to this work is the study of *multiple disjoint repair groups.*

▶ **Definition 1.** Given a code $\mathcal{C} \subset \Sigma^N$, we say that a set $S \subset \{1, \ldots, N\}$ is a *repair group* for $i \in \{1, \ldots, N\}$ if $i \notin S$, and if there is some function $g : \Sigma^{|S|} \to \Sigma$ so that $g(c|_S) = c_i$ for all $c \in \mathcal{C}$. That is, the codeword symbols indexed by S uniquely determine the symbol indexed by i.

▶ **Definition 2.** We say that $\mathcal{C}$ has the s-Disjoint-Repair-Group Property (s-DRGP) if for every $i \in \{1, \ldots, N\}$, there are s disjoint repair groups $S_1^{(i)}, \ldots, S_s^{(i)}$ for i.

In the context of LRCs, the parameter s is called the *availability* of the code. An LRC with availability s is not exactly the same as a code with the s-DRGP (the difference is that, in Definition 2, there is no mention of the size Q of the repair groups), but it turns out to be deeply related; it is also directly related to other notions of locality in distributed storage (like batch codes), as well as in cryptography (like PIR codes). We will review some of this work below, and we point the reader to [24] for a survey of batch codes, PIR codes, and their connections to LRCs and the s-DRGP.

While originally motivated for small s, as we will see below, the s-DRGP is interesting (and has already been implicitly studied) for a wide range of s, from $O(1)$ to $\Omega(N)$. For

[1] In some works, the guarantee holds for information symbols only, rather than for all codeword symbols; we stick with all symbols here for simplicity of exposition.

$s = o(N)$, we can hope for codes with very high rate, approaching 1; the question is how fast we can hope for this rate to approach 1. More formally, if $K = \log_{|\Sigma|} |\mathcal{C}|$, then the rate is K/N, and we are interested in how the gap $N - K$ behaves with N and s. We will refer to the quantity $N - K$ as the *co-dimension* of the code; when $\mathcal{C}$ is linear (that is, when $\Sigma = \mathbb{F}$ is a finite field and $\mathcal{C} \subseteq \mathbb{F}^N$ is a linear subspace), then this is indeed the co-dimension of $\mathcal{C}$ in $\mathbb{F}^N$. The main question we seek to address in this paper is the following.

▶ **Question 1.** *For a given s and N, what is the smallest codimension $N - K$ of any code with the s-DGRP? In particular, how does this quantity depend on s and N?*

We know a few things about Question 1, which we survey below. However, there are many things about this question which we still do not understand. In particular, the dependence on s is wide open, and this dependence on s is the focus of the current work. Below, we survey the state of Question 1 both from the LRC end (when s is small) and the LCC end (when s is large).

The s-DRGP when s is small

In [28, 21, 25, 27], the s-DRGP was explicitly considered, with a focus on small s ($s = 2$ is of particular interest). In those works, some bounds on the rate and distance of codes with the s-DRGP were derived (some of them in terms of the locality Q). However, for larger s, these bounds degrade. More precisely, [28, 21] establish bounds on $N - K$ in terms of Q, s, and the distance of the code, but as s grows these are not much stronger than the Singleton bound. The results of [25, 27] give an upper bound on the rate of a code in terms of Q and s. One corollary is that the rate satisfies $K/N \leq (s+1)^{-1/Q}$; if we are after high-rate codes, this implies that we must take $Q = \Omega(\ln(s+1))$, and this implies that the codimension $N - K$ must be at least $\Omega(N \ln(s)/Q)$.

A similar notion to the s-DRGP was introduced in [8], with the application of *Private Information Retrieval* (PIR). PIR schemes are an important primitive in cryptography, and they have long been linked to constant-query LCCs. In [8], PIR was also shown to be related to the s-DRGP. The work [8] introduces *PIR codes*, which enable PIR schemes with much less storage overhead. It turns out that the requirement for PIR codes is very similar to the s-DRGP.[2]

In the context of PIR codes [8, 6], there are constructions of s-DRGP codes with $N - K \leq O(s\sqrt{N})$. For $s = 2$, this is known to be tight, and there is a matching lower bound [20]. However, it seems difficult to use this lower bound technique to prove a stronger lower bound when s is larger (possibly growing with N).

The s-DRGP when s is large

As we saw above, when s is small then the s-DRGP is intimately related to LRCs, PIR codes and batch codes. On the other end of the spectrum, when s is large (say, $\Omega(N)$ or $\Omega(N^{1-\varepsilon})$) then it is related to LCCs.

When $s = \Omega(N)$, then the s-DRGP is in fact *equivalent* to a constant-query LCC (that is, an LCC as described above, where the number of queries to $\tilde{c}$ is $O(1)$). The fact that the $\Omega(N)$-DRGP implies a constant-query LCC is straightforward: the correction algorithm to recover c_i is to choose a random j in $\{1, \ldots, s\}$ and use the repair group $S_j^{(i)}$ to recover c_i.

[2] The only difference is that PIR codes only need to recover information symbols, but possibly with non-systematic encoding.

Since in expectation the size of $S_j^{(i)}$ is constant, we can restrict our attention only to the constant-sized repair groups. Then, with some constant probability none of the indices in $S_j^{(i)}$ will be corrupted, and this success probability can be amplified by independent repetitions. The converse is also true [16, 29], and any constant-query LCC has the s-DRGP for $s = \Omega(n)$; in fact, this connection is one of the few ways we know how to get lower bounds on LCCs.

When s is large, but not as large as $\Omega(N)$, there is still a tight relationship with LCCs. By now we know of several high-rate ($(1-\alpha)$, for any constant α) LCCs with query complexity $Q = N^{\varepsilon}$ for any $\varepsilon > 0$ [17, 12, 13] or even $Q = N^{o(1)}$ [18]. It is easy to see[3] that any LCC with query complexity Q has the s-DGRP for $s = \Omega(N/Q)$. Thus, these codes immediately imply high-rate s-DRGP codes with $s = \Omega(N^{1-\varepsilon})$ or even larger. (See also [1]). Conversely, the techniques of [13, 18] show how to take high-rate linear codes with the s-DGRP for $s = \Omega(N^{1-\varepsilon})$ and produce high-rate LCCs with query complexity $O(N^{\varepsilon'})$ (for a different constant ε').

These relationships provide some bounds on the codimension $N - K$ in terms of s: from existing lower bounds on constant-query LCCs [29], we know that any code with the s-DGRP and $s = \Omega(N)$ must have vanishing rate. On the other hand from high-rate LCCs, there exist s-DGRP codes with $s = \Omega(N^{1-\varepsilon})$ and with high rate. However, these techniques do not immediately given anything better than high (constant) rate, while in Question 1 we are interested in precisely controlling the co-dimension $N - K$.

The s-DRGP when s is intermediate

The fact that the s-DRGP interpolates between the LRC setting for small s and the LCC setting for large s motivates the question of the s-DGRP for intemediate s, say $s = \log(N)$ or $s = N^c$ for $c < 1/2$. Our goal is to understand the answer to Question 1 for intermediate s.

We have only a few data points to answer this question. As mentioned above, the constructions of [8, 6] show that there are codes with $N - K \leq s\sqrt{N}$ for $s \leq \sqrt{N}$. However, the best general lower bounds known [20, 27] can only establish $N - K \geq \max\left\{\sqrt{2N}, N - \frac{N}{(s+1)^{1/Q}}\right\}$. Above, we recall that Q is a parameter bounding the size of the repair groups; in order for the second term above (from [27]) to be $o(N)$, we require $Q \gg \ln(s+1)$; in this case, the second bound on the codimension reads $N - K \geq \Omega(N\ln(s)/Q)$. As the size of the repair groups Q may in general be as large as N/s, in our setting this second bound gives better dependence on s, but worse dependence on N.

The upper bound of $s\sqrt{N}$ is not tight, at least for large s. For $s = \sqrt{N}$, there are several classical constructions which have the s-DRGP and with $N - K = \Theta(N^{\log_4(3)})$; for example, this includes affine geometry codes and/or codes constructed from difference sets (see [2], [19], or [12] – we will also recover these in Corollary 15). Notice that this is much better than the upper bound of $N - K \leq s\sqrt{N}$, which for $s = \sqrt{N}$ would be trivial.

However, other than these codes, before this work we did not know of any constructions for

[3] Indeed, suppose that $\mathcal{C}$ is an LCC with query complexity Q and error tolerance δ, and let $s = \delta N/Q$. In order to obtain s disjoint repair groups for a symbol c_i from the LCC guarantee, we proceed as follows. First, we make one (randomized) set of queries to c; this gives the first repair group. Continuing inductively, assume we have found $t \leq s$ disjoint repair groups already, covering a total of at most $tQ < \delta N$ symbols. To get the $t+1$'st set of queries, we again choose at random as per the LCC requirement. These queries may not be disjoint from the previous queries, but the LCC guarantee can handle errors (and hence erasures) in up to δN positions, so it suffices to query the points which have not been already queried, and treat the already-queried points as unavailable. We repeat this process until t reaches $s = \delta N/Q$.

$s \ll \sqrt{N}$ which beat the bounds in [8, 6] of $N - K \leq s\sqrt{N}$.[4] One of the main contributions of this work is to give a construction with $s = N^{1/4}$, which achieves codimension $N - K = N^{0.714}$. Notice that the bound of $s\sqrt{N}$ would be $N^{0.75}$ in this case, so this is a substantial improvement. We remark that we do not believe that our construction is optimal, and unfortunately we don't have any deep insight about the constant 0.714. Rather, we stress that the point of this work is to (a) highlight the fact that the $s\sqrt{N}$ bound can be beaten for $s \ll \sqrt{N}$, and (b) highlight our techniques, which we believe may be of independent interest.

1.2 Lifted codes, and our construction

Our construction is based on the *lifted codes* of Guo, Kopparty and Sudan [12]. The original motivation for lifted codes was to construct high-rate LCCs, as described above. However, since then they have found several other uses, for example list-decoding and local-list-decoding [11]. The codes are based on multivariate polynomials, and we describe them below. Suppose that $\mathcal{F} \subseteq \mathbb{F}_q[X, Y]$ is a collection of bivariate polynomials over a finite field $\mathbb{F}_q$ of order q. This collection naturally gives rise to a code $\mathcal{C} \subseteq \mathbb{F}^{q^2}$:

$$\mathcal{C} = \left\{ \langle P(x, y) \rangle_{(x,y) \in \mathbb{F}_q^2} \, : \, P \in \mathcal{F} \right\}. \tag{1}$$

Above, we assume some fixed order on the elements of $\mathbb{F}_q^2$, and by $\langle P(x, y) \rangle_{(x,y) \in \mathbb{F}_q^2}$, we mean the vector in $\mathbb{F}_q^{q^2}$ whose entries are the evaluations of P in this prescribed order. For example, a bivariate Reed-Muller code is formed by taking $\mathcal{F}$ to be the set of all polynomials of total degree at most d.

One nice property of Reed-Muller codes is their locality. More precisely, suppose that $P(X, Y)$ is a bivariate polynomial over $\mathbb{F}_q$ of total degree at most d. For an affine line in $\mathbb{F}_q^2$, parameterized as $L(T) = (\alpha T + \beta, \gamma T + \delta)$, we can consider the *restriction* $P|_L$ of P to L, given by

$$P|_L(T) := P(\alpha T + \beta, \gamma T + \delta) \mod T^q - T,$$

where we think of the above as a polynomial of degree at most $q - 1$. It is not hard to see that if P has total degree at most d, then $P|_L(T)$ also has degree at most d; in other words, it is a univariate Reed-Solomon codeword. This property – that the restriction of any codeword to a line is itself a codeword of another code – is extremely useful, and has been exploited in coding theory since Reed's majority logic decoder in the 1950's [22]. A natural question is whether or not there exist any bivariate polynomials $P(X, Y)$ *other* than those of total degree at most d which have this property. That is, are there polynomials which have high degree, but whose restrictions to lines are always low-degree? In many settings (for example, over the reals, or over prime fields) the answer is no. However, the insight of [12] is that there are settings – high degree polynomials over small-characteristic fields – for which the answer is yes.

This motivates the definition of *lifted codes*, which are multivariate polynomial evaluation codes, all of whose restrictions to lines lie in some other base code. Guo, Kopparty and

[4] We note that there have been some works in the intermediate-s parameter regime which can obtain excellent locality Q but are not directly relevant for Question 1. In particular, the work of [21] gives a construction of s-DRGP codes with $s = \Theta(K^{1/3-\varepsilon})$ and $Q = \Theta(K^{1/3})$ for arbitarily small constant ε; while this work obtains a smaller Q than we will eventially obtain (our results will have $Q \sim \sqrt{N}$), they are only able to establish high (constant) rate codes, and thus do not yield tight bounds on the co-dimension. The work of [3] gives constructions of high-rate fountain codes which have $s, Q = \Theta(\log(N))$. As these are rateless codes, again they are not directly relevant to Question 1.

Sudan showed that, in the case above, not only do these codes exist, but in fact they may have rate much higher than the corresponding Reed-Muller code.

Lifted codes very naturally give rise to codes with the s-DRGP. Indeed, consider the bivariate example above, with $d = q - 2$. That is, $\mathcal{C}$ is the set of codewords arising from evaluations of functions P that have the property that for all lines $L : \mathbb{F}_q \to \mathbb{F}_q^2$, $\deg(P|_L) \leq q - 2$. The restrictions then lie in the parity-check code: we always have $\sum_{t \in \mathbb{F}_q} P|_L(t) = 0$. Thus, for every coordinate of a codeword in $\mathcal{C}$ – which corresponds to an evaluation point $(x, y) \in \mathbb{F}_q^2$ – there are q disjoint repair groups for this symbol, corresponding to the q affine lines through (x, y).

However, it's not obvious how to use these codes to obtain the s-DRGP for $s \ll \sqrt{N}$; increasing the number of variables causes s to grow, and this is the approach taken in [12] to obtain high-rate LCCs. Since we are after smaller s, we take a different approach. We stick with bivariate codes, but instead of requiring that the functions $P \in \mathcal{F}$ restrict to low-degree polynomials on *all* affine lines L, we make this requirement only for *some* lines. This allows us to achieve the s-DRGP (if there are s lines through each point), while still being able to control the rate.

While special cases of this idea – notably tensor codes – have been considered before, allowing more complicated sets of lines requires some new machinery, and we hope that this machinery may be useful more generally. In the next section, we will set up our notation and give an outline of this approach, after a brief review of the notation we will use throughout the paper.

1.3 Outline

Next, in Section 2, we define *partially lifted codes*, and give a technical overview of our approach. This approach consists of two parts. The first is a general framework for understanding the dimension of partially lifted codes of a certain form, which we then discuss more in Section 3. The second part is to instantiate this framework, which we do in Section 4. This gives rise to the s-DRGP code with $s = N^{1/4}$ described above. Due to space constraints, we omit many details from this extended abstract, and refer the reader to the full version of the paper [9].

2 Technical Overview

In this section, we give a high-level overview of our construction and approach. We begin with some basic definitions and notation.

2.1 Notation and basic definitions

We study linear codes $\mathcal{C} \subseteq \mathbb{F}_q^N$ of block length N over an alphabet of size q. We will always assume that $\mathbb{F}_q$ has characteristic 2, and write $q = 2^\ell$. (We note that this is not strictly necessary for our techniques to apply – the important thing is only that the field is of relatively small characteristic – but it simplifies the analysis, and so we work in this special case).

The specific codes $\mathcal{C}$ that we consider are *polynomial evaluation codes*. Formally, let $\mathcal{F}$ be a collection of m-variate polynomials over $\mathbb{F}_q$. Letting $N = q^m$, we may identify $\mathcal{F}$ with a code $\mathcal{C} \subseteq \mathbb{F}_q^N$ as in (1); we assume that there is some fixed ordering on the elements of $\mathbb{F}_q^m$ to make this well-defined. For a polynomial $P \in \mathbb{F}_q[X_1, \ldots, X_m]$, we write its corresponding codeword as

$$\mathsf{eval}(P) = \langle P(x_1, \ldots, x_m) \rangle_{(x_1,\ldots,x_m)\in\mathbb{F}_q^m} \in \mathcal{C}.$$

We will only focus on $m = 1, 2$, as we consider the restriction of bivariate polynomial codes to lines, which results in univariate polynomial codes. Formally, a (parameterization of an) *affine line* is a map $L : \mathbb{F}_q \to \mathbb{F}_q^2$, of the form $L(T) = (\alpha T + \beta, \gamma T + \delta)$ for $\alpha, \beta, \gamma, \delta \in \mathbb{F}_q$. We say that two parameterizations L, L' are *equivalent* if the result in the same line as a set: $\{L(t) : t \in \mathbb{F}_q\} = \{L'(t) : t \in \mathbb{F}_q\}$. We denote the restriction of a polynomial $P \in \mathbb{F}_q[X, Y]$ to L by $P|_L$:

▶ **Definition 3.** For a line $L : \mathbb{F}_q \to \mathbb{F}_q^2$ with $L(T) = (L_1(T), L_2(T))$, and a polynomial $P : \mathbb{F}_q^2 \to \mathbb{F}_q$, we define the *restriction* of P on L, denoted $P|_L : \mathbb{F}_q \to \mathbb{F}_q$, to be the unique polynomial of degree at most $q - 1$ so that $P|_L(T) = P(L_1(T), L_2(T))$.

We note that the definition above makes sense, because all functions $f : \mathbb{F}_q \to \mathbb{F}_q$ can be written as polynomials of degree at most $q - 1$ over $\mathbb{F}_q$; in this case, we have $P|_L(T) = P(L_1(T), L_2(T)) \mod (T^q - T)$.

▶ Remark 1. Throughout this paper, all polynomials will be considered mod $T^q - T$, although we will frequently drop this notation for ease of reading.

Finally, we'll need some tools for reasoning about integers and their binary expansions.

▶ **Definition 4.** Let $m < q$ be a positive integer. If $m = \sum_{i=0}^{\ell-1} m_i 2^i$, where $m_i \in \{0, 1\}$, then we let $B(m) = \{i \in \{0, ..., \ell - 1\} \mid m_i = 1\}$. That is, $B(m)$ is the set of indices where the binary expansion of m has a 1.

▶ **Definition 5.** For any two integers $m, n < q$, we say that m lies in the 2-shadow of n, denoted $m \leq_2 n$, if $B(m) \subseteq B(n)$. Equivalently, letting $m = \sum_{i=0}^{\ell-1} m_i 2^i$ and $n = \sum_{i=0}^{\ell-1} n_i 2^i$, we write $m \leq_2 n$ if for all $i \in \{0, ..., \ell - 1\}$, whenever $m_i = 1$ then also $n_i = 1$.

The reason that we are interested in 2-shadows is because of Lucas' Theorem.

▶ **Theorem 6** (Lucas' Theorem). *For any $m, n \in \mathbb{Z}$, $\binom{m}{n} \equiv 0 \mod 2$ exactly when $m \not\leq_2 n$.*

Finally, for integers a, b, s, we will say $a \equiv_s b$ if a is equal to b modulo s. For a positive integer n, we use $[n]$ to denote the set $[n] = \{0, \ldots, n - 1\}$.

2.2 Partially lifted codes

With the preliminaries out of the way, we proceed with a description of our construction and techniques. As alluded to above, our codes will be bivariate polynomial codes, which are "partial lifts" of parity check codes.

▶ **Definition 7.** Let $\mathcal{F}_0 \subseteq \mathbb{F}_q[T]$ be a collection of univariate polynomials, and let $\mathcal{L}$ be a collection of parameterizations of affine lines $L : \mathbb{F}_q \to \mathbb{F}_q^2$. We define the *partial lift* of $\mathcal{F}_0$ with respect to $\mathcal{L}$ to be the set

$$\mathcal{F} = \{P \in \mathbb{F}_q[X, Y] : \forall P \in \mathcal{F}, \forall L \in \mathcal{L}, P|_L \in \mathcal{F}_0\}.$$

We make a few remarks about Definition 7 before proceeding.

▶ Remark 2 (Equivalent lines). We remark that the definition above allows $\mathcal{L}$ to be a collection of *parameterizations* of lines. A priori, it is possible that equivalent parameterizations may behave very differently with respect to $\mathcal{F}_0$, and it is also possible to include several equivalent parameterizations in $\mathcal{L}$. In this work, $\mathcal{F}_0$ will always be affine-invariant (in particular, it will just be the set of polynomials of degree strictly less than $q-1$), and so if L and L' equivalent, then $P|_L \in \mathcal{F}_0$ if and only if $P|_{L'} \in \mathcal{F}_0$. Thus, these issues won't be important for this work.

▶ Remark 3 (Why only bivariate lifts?). This definition works just as well for m-variate partial lifts, and we hope that further study will explore this direction. However, as all of our results are for bivariate codes, we will stick to the bivariate case to avoid having to introduce another parameter.

Let $\mathcal{F}_0 := \{P \in \mathbb{F}_q[X], \deg(P) < q-1\}$. Then it is not hard to see that the code $\mathcal{C}_0 = \{\mathsf{eval}(P) : P \in \mathcal{F}_0\}$ is just the parity-check code, $\mathcal{C}_0 = \left\{c \in \mathbb{F}_q^q : \sum_{i=1}^q c_i = 0\right\}$. Indeed, for any $d < q-1$, we have $\sum_{x \in \mathbb{F}_q} x^d = 0$.

We will construct codes with the s-DRGP by considering codes that are partial lifts of $\mathcal{F}_0$. We first observe that such codes, with an appropriate set of lines $\mathcal{L}$, will have the s-DRGP. Indeed, suppose we wish to recover a particular symbol, given by $P(x,y)$ for $(x,y) \in \mathbb{F}_q^2$. Let $L^{(1)}, \ldots, L^{(s)} \in \mathcal{L}$ be s distinct (non-equivalent) lines that pass through (x,y); say they are parameterized so that $L^{(j)}(0) = (x,y)$. Then the s disjoint repair groups are the sets indices corresponding to $S_j := \{L^{(j)}(t) : t \in \mathbb{F}_q \setminus \{0\}\}$. For any P in the partial lift of $\mathcal{F}_0$, we have $P|_L(0) = \sum_{t \in \mathbb{F}_q \setminus \{0\}} P|_L(t)$, which means that $P(x,y) = \sum_{(a,b) \in S_j} P(a,b)$. That is, $P(x,y)$ can be recovered from the coordinates of $\mathsf{eval}(P)$ indexed by S_j, as desired. Finally we observe that the S_j are all disjoint, as the lines are all distinct, and intersect only at (x,y). We summarize the above discussion in the following observation.

▶ **Observation 8.** *Suppose that $\mathcal{F}_0 = \{P \in \mathbb{F}_q[T] : \deg(P) < q-1\}$, and let $\mathcal{L}$ be any collection of parameterizations of affine lines so that every point in $\mathbb{F}_q^2$ is contained in at least s non-equivalent elements of $\mathcal{L}$. Let $\mathcal{F}$ be the bivariate partial lift of $\mathcal{F}_0$ with respect to $\mathcal{L}$. Then the code $\mathcal{C} \subseteq \mathbb{F}_q^{q^2}$ corresponding to $\mathcal{F}$ is a linear code with the s-DRGP.*

To save on notation later, we say that a polynomial $P : \mathbb{F}_q^2 \to \mathbb{F}_q$ *restricts nicely* on a line $L : \mathbb{F}_q \to \mathbb{F}_q^2$ if $P|_L$ has degree strictly less than $q-1$. Thus, to define our construction, we have to define the collection $\mathcal{L}$ of lines used in Definition 7. We will actually develop a framework that can handle a family of such collections, but for intuition in this section, let us just consider lines $L(T) = (T, \alpha T + \beta)$ where α lives in a multiplicative subgroup G_s of $\mathbb{F}_q^*$ of size s, and $\beta \in \mathbb{F}_q$. That is, we are essentially restricting the slope of the lines to lie in a multiplicative subgroup. It is not hard to see that every point $(x,y) \in \mathbb{F}_q^2$ has s non-equivalent lines in $\mathcal{L}$ that pass through it.

Following Observation 8, the resulting code will immediately have the s-DRGP. The only question is, what is the rate of this code? Equivalently, we want to know:

▶ **Question 2.** *How many polynomials $P \in \mathbb{F}_q[X,Y]$ have $\deg(P|_L) < q-1$ for all $L \in \mathcal{L}$, where $\mathcal{L}$ is as described above?*

In [12], Guo, Kopparty and Sudan develop some machinery for answering this question when $\mathcal{L}$ is the set of all affine lines. What they show in that work is that in fact the (fully) lifted code is affine-invariant, and is equal to the span of the monomials $P(X,Y) = X^aY^b$ so that $\deg(P|_L) < q-1$ for all affine lines L. We might first hope that this is the case for partial lifts – but then upon reflection we would immediately retract this hope, because it turns out that we do not get any more monomials this way: Theorem 13 establishes that if a monomial restricts nicely on even one line of the form $(T, \alpha T + \beta)$ (for nonzero α, β), then in fact it restricts nicely on *all* such lines. In fact, the partial lift is not in general affine-invariant, and this is precisely where we are able to make progress. More precisely, there may be polynomials $P(X,Y)$ of the form

$$P(X,Y) = X^{a_1}Y^{b_1} + X^{a_2}Y^{b_2} \tag{2}$$

which are contained in the partial lift $\mathcal{F}$, but so that $X^{a_1}Y^{b_1}, X^{a_2}Y^{b_2} \notin \mathcal{F}$. This gives us many more polynomials to use in a basis for $\mathcal{F}$ than just the relevant monomials, and allows us to construct families $\mathcal{F}$ of larger dimension.

▶ Remark 4 (Breaking affine invariance). We emphasize that breaking affine-invariance is a key departure from [12]. In some sense, it is not surprising that we are able to make progress by doing this: the assumption of affine-invariance is one way to prove *lower bounds* on locality [4, 5]. This is also where our techniques diverge from those of [12]. Because of their characterization of affine-invariant codes, that work focused on understanding the dimension of the relevant set of monomials. This is not sufficient for us, and so to get a handle on the dimension of our constructions, we must study more complicated polynomials. This may seem daunting, but we show – perhaps surprisingly – that one can make a great deal of progress by considering only the additional "more complicated" polynomials of the form (2), which are arguably the simplest of the "more complicated" polynomials.

In order to obtain a lower bound on the dimension of $\mathcal{F}$, our strategy get a handle on the dimension of the space of these binomials (2). If we can show that there are many linearly independent such binomials, then the answer to Question 2 must be "lots."

Following this strategy, we examine binomials of the form (2), and we ask, for which a_1, b_1, a_2, b_2 and which $L(T) = (T, \alpha T + \beta)$ does $P(X,Y)$ restrict nicely? Our main tool is Lucas's Theorem (Theorem 6), which was also used in [12]. To see why this is useful, consider the restriction of a monomial $P(X,Y) = X^aY^b$ to a line $L(T) = (T, \alpha T + \beta)$. We obtain

$$P|_L(T) = T^a\left(\alpha T + \beta\right)^b = \sum_{i \le b} \binom{b}{i} \alpha^i \beta^{b-i} T^{a+i}.$$

Above, the binomial coefficient $\binom{b}{j}$ is shorthand for the sum of 1 with itself $\binom{b}{j}$ times. Thus, in a field of characteristic 2, this is either equal to 1 or equal to 0; Lucas's theorem tells us which it is. This means that our question reduces to asking, when does the coefficient of T^{q-1} vanish? The above gives us an expression for this coefficient, and allows us to compute an answer, in terms of the binary expansions of a and b.

So far, this is precisely the approach of [12]. From here, we turn to the binomials of the form (2). When do these restrict nicely? As above, we may compute the coefficient of the T^{q-1} term and examine it. Fortunately, when the set of lines $\mathcal{L}$ is chosen as above, the number of linearly independent binomials that restrict nicely ends up having a nice expression, in terms of the number of non-empty equivalence classes of a particular relation defined by the binary expansion of the numbers $1, \ldots, q-1$; this is our main technical theorem (Theorem 12, which is proved in Section 3.2).

The approach of Section 3.2 holds for more general families than the $\mathcal{L}$ described above; instead of taking α in a multiplicative subgroup of $\mathbb{F}_q^*$, we may alternately restrict β, or restrict both. However, numerical calculations indicated that the choice above (where α is in a multiplicative subgroup of order s) is a good one, so for our construction we make this choice and we focus on that for our formal analysis in Section 4.

In order to get our final construction and obtain the results advertised above, it suffices to count these equivalence classes. For the result advertised in the introduction, we choose the order of the multiplicative subgroup to be $s = 2^{\ell/2} - 1 = \sqrt{q} - 1$. Then, we use an inductive argument in Section 4 to count the resulting equivalence classes, obtaining the bounds advertised above. More precisely, we obtain the following theorem.

▶ **Theorem 9.** *Suppose that $q = 2^\ell$ for even ℓ, and let $N = q^2 - 1$. There is a linear code $\mathcal{C}$ over $\mathbb{F}_q$ of length N and dimension*

$$K \geq N - O(N^{.714})$$

which has the s-DRGP for $s = \sqrt{q} - 2 = (N+1)^{1/4} - 1$.

▶ Remark 5 (Puncturing at the origin). We note that the statement of the theorem differs slightly from the informal description above; in our analysis, we will puncture the origin, and ignore lines that go through the origin; that is, our codes will have length $q^2 - 1$, rather than q^2, and the number of lines through every point will be $s - 1$, rather than s, as it makes the calculations somewhat easier and does not substantially change the results.

2.3 Discussion and open questions

Before we dive into the technical details in Section 3, we close the front matter with some discussion of open questions left by our work and our approach. We view the study of the s-DRGP for intermediate s to be an important step in understanding locality in general, since the s-DRGP nicely interpolates between the two extremes of LRCs and LCCs. When $s = 2$, we completely understand the answer to Question 1. However, by the time s reaches $\Omega(N)$, this becomes a question about the best rate of constant-query LCCs, which is a notoriously hard open problem. It is our hope that by better understanding the s-DRGP, we can make progress on these very difficult questions.

The main question left by our work is Question 1, which we do not answer. What is the correct dependence on s in the codimension of codes with the s-DRGP? We have shown that it is not $s\sqrt{N}$, even for $s \ll \sqrt{N}$. However, we have no reason to believe that our construction is optimal.

Our work also raises questions about partially lifted codes. These do not seem to have been studied before. The most immediate question arising from our work is to improve or generalize our approach; in particular, is our analysis tight? Our approach proceeds by counting the binomials of the form (2). This is in principle lossy, but empirical simulations suggest that at least in the setting of Theorem 9, this approach is basically tight. Are there situations in which this is not tight? Or can we prove that it is tight in any situation? Finally, are there other uses of partially lifted codes? As with lifted codes, we hope that these prove useful in a variety of settings.

3 Framework

As discussed in the previous section, the proof of Theorem 9 is based on the partially lifted codes of Definition 7. In this section, we lay out the partially lifted codes we consider, as well as the basic tools we need to analyze them. As before, we say that a polynomial $P : \mathbb{F}_q^2 \to \mathbb{F}_q$ *restricts nicely* to a line $L : \mathbb{F}_q \to \mathbb{F}_q^2$ if $P|_L$ has degree strictly less than $q - 1$. We will consider partial lifts of the parity-check code with respect to a collection of affine lines $\mathcal{L}$; reasoning about the rate of this code will amount to reasoning about the polynomials which restrict nicely to lines in $\mathcal{L}$. To ease the computations, we will form our family $\mathcal{L}$ out of lines that have a simple parameterization:

▶ **Definition 10.** We say a line $L : \mathbb{F} \to \mathbb{F}^2$ is *simple* if it can be written in the form $L(T) = (T, \alpha T + \beta)$, with $\alpha, \beta \neq 0$.

Notice that this rules out lines through the origin. At the end of the day, we will pucture our code at the origin to achieve our final result. Note also that no two simple parameterizations of lines are equivalent to each other (that is, they form distinct lines as sets), so as we go forward, we may apply Observation 8 without worry of the repair groups coinciding.

We consider a family of constructions, indexed by parameters s and t, so that $s, t \mid q - 1$. This family will be the partial lift with respect to the following set of simple lines.

▶ **Definition 11.** Let $s, t \mid q - 1$, and let $G_s, G_t \subseteq \mathbb{F}_q^*$ be multiplicative subgroups of $\mathbb{F}_q^*$ of orders s and t, respectively. That is, $G_s = \left\{x \in \mathbb{F}_q^* \,:\, x^s = 1\right\}$ and $G_t = \left\{x \in \mathbb{F}_q^* \,:\, x^t = 1\right\}$. Then we define $\mathcal{L}_{s,t}$ to be the family of simple lines

$$\mathcal{L}_{s,t} = \{L(T) = (T, \alpha T + \beta) \,:\, \alpha \in G_s, \beta \in G_t\}\,.$$

For the rest of the paper, we will study the following construction, for various choices of s and t.

▶ **Construction 1.** *Let $\mathcal{L}_{s,t}$ be as in Definition 11 for $s, t \mid q - 1$, and let $\mathcal{F}_0$ be the set of univariate polynomials of degree strictly less than $q - 1$. Define $\mathcal{F}_{s,t}$ to be the partial lift of $\mathcal{F}_0$ with respect to $\mathcal{L}_{s,t}$.*

Our main theorem, which we will prove in the rest of this section, is a characterization of the dimension of $\mathcal{F}_{s,t}$ as in Construction 1. (We recall the definition of $\leq_2$ from Definition 5 above).

▶ **Theorem 12.** *Suppose that $s, t \mid q - 1$. For nonnegative integers $i < s, j < t$, define*

$$\begin{aligned} e(s,t) = |\{(i,j) \ &: i < s, \text{ and } j < t, \\ &\quad \text{so that there is some } m, n \in [q]^2 \text{ with } m \equiv_s i, n \equiv_t j, \text{ and } n \leq_2 m\}|\,. \end{aligned}$$

Then the dimension of $\mathcal{F}_{s,t} \subseteq \mathbb{F}_q[X,Y]$ is at least

$$\dim(\mathcal{F}_{s,t}) \geq q^2 - e(s,t).$$

Theorem 12 may seem rather mysterious. The expression $e(s,t)$ comes up in counting the number of binomials of the form (2) the restrict nicely on lines in $\mathcal{L}_{s,t}$. We omit the full proof of Theorem 12 in this extended abstract, but we will sketch the outline in Section 3.2.

The reason that Theorem 12 is useful is that for some s and t, it turns out to be possible to get a very tight handle on $e(s,t)$, which leads to the quantitative result in Thorem 9. For now, we focus on proving Theorem 12. Our starting point is the work of [12]; we summarize the relevant points below in Section 3.1.

3.1 Basic Setup: Lucas' Theorem and Monomials

In [12], Guo, Kopparty and Sudan give a characterization of lifted codes. In our setting, their work shows that when the set $\mathcal{L}$ is the set of *all* affine lines, then the lifted code $\mathcal{F}$ is affine invariant and in fact is equal to the span of the *monomials* which restrict nicely. In the case where the number of variables is large, or the base code $\mathcal{F}_0$ is more complicated than a parity-check code, [12] provides some bounds, but it seems quite difficult to get a tight characterization of these monomials. However, for bivariate lifts of the parity-check code, it is actually possible to completely understand the situation, and this was essentially done in [12]. We review their approach here.

First, we use Lucas' Theorem (Theorem 6) to characterize which monomials X^aY^b restrict nicely to simple lines. Theorem 13 follows from the analysis in [12]; we refer the reader to the full version of this paper [9] for a direct proof.

▶ **Theorem 13.** *Suppose $a + b < 2(q - 1)$ and let $P(X, Y) = X^aY^b$. Then for all simple lines $L(T) = (T, \alpha T + \beta)$, $P|_L$ has degree $< q - 1$ if and only if $q - 1 - a \not\leq_2 b$. Further, if $q - 1 - a \leq_2 b$, then $P|_L$ is a degree $q - 1$ polynomial with leading coefficient $\alpha^{-a}\beta^{b+a}$*

Theorem 13 implies that whether a monomial $P(X, Y) = X^aY^b$ restricts nicely to a simple line L is independent of the choice of L. Thus it makes sense to consider this a property of the monomial itself.

▶ **Definition 14.** We say that a monomial $P(X, Y) = X^aY^b$ with $0 \leq a, b \leq q - 1$ is *good* if it restricts nicely on all simple lines.

▶ Remark 6 (The special case of $X^{q-1}Y^{q-1}$). In Theorem 13, we required $a + b < 2(q - 1)$, which does not cover the monomial $P_*(X, Y) = X^{q-1}Y^{q-1}$. However, in Definition 14, we allow $a = b = q - 1$, and in fact according to this definition $P_*(X, Y)$ is good; we will treat it that way in this work, even though it would not be considered good in the analysis of [12]. (In their language, P_* does not live in the lift of the degree set $\{0, \ldots, q - 2\}$).

Theorem 13 implies (see [9]) that there are $q^2 - 3^\ell + 1$ good monomials. This allows us to recover the codes of Theorem 1.2 in [12] up to the technicalities about simple lines vs. all lines. Following Observation 8, these codes have the s-DRGP for $s = q - 1$; indeed, there are $q - 1$ simple lines through every non-zero point of $\mathbb{F}_q^2$. The dimension of these codes is at least the number of monomials that they contain (indeed, all monomials are linearly independent), which by the above is at least $q^2 - 3^\ell + 1 = (N + 1) - (N + 1)^{\log_4(3)} + 1$.

▶ **Corollary 15** (Implicit in [12]). *There are codes linear $\mathcal{C}$ over $\mathbb{F}_q$ of length $N = q^2 - 1$ with dimension $K \geq N + 2 - (N + 1)^{\log_4(3)}$ which have the s-DRGP for $s = q - 1 = \sqrt{N + 1} - 1$.*

We note that this recovers the results of one of the classical constructions of the s-DRGP for $s = \sqrt{N}$ mentioned in the introduction (and this is not an accident: these codes are in fact the same as affine geometry codes). In the next section, we show how to use the relaxation to partial lifts in order to create codes with the s-DRGP for $s \ll \sqrt{N}$.

3.2 Partially lifted codes

In this section we extend the analysis above to partial lifts. The work of [12] characterizes the polynomials which restrict nicely on all lines $L : \mathbb{F}_q \to \mathbb{F}_q^2$: they show that this is exactly the span of the good monomials (except the special monomial P_* of Remark 6, which restricts to degree lower than $q - 1$ only on *simple* lines). However, since our goal is to obtain codes with the s-DRGP for $s \ll \sqrt{N}$, increasing the dimension while decreasing s, we would like to allow for more polynomials.

Thus, as in Definition 7, we will consider polynomials which restrict nicely only on some particular subset $\mathcal{L}$ of simple lines. We would like to find a subset $\mathcal{L}$ such that the space of polynomials which restrict nicely on all lines in $\mathcal{L}$ has large degree. Additionally, we would like to guarantee the s-DRGP by ensuring that, for every point (x, y), there are many lines in $\mathcal{L}$ that pass through (x, y). Relaxing requirements in this manner will allow us to get codes with good rate and locality trade-offs.

Theorem 13 shows that if a monomial restricts nicely on one simple line, it will restrict nicely on all simple lines. This means that in order to find a larger space of polynomials, we cannot only consider monomials. Towards this end, we will consider *binomials* of the form

$$P(X,Y) = X^{a_1}Y^{b_1} + X^{a_2}Y^{b_2}. \tag{3}$$

That is, we will look only at binomials with both coefficients equal to 1.

We note that this ability to extend beyond monomials is possible crucially because our partially lifted codes are not affine-invariant. While affine-invariance allowed [12] to get a beautiful characterization of (fully) lifted codes, it also greatly restricts the flexibility of these codes. By breaking affine-invariance, we also break some of the rigidity of these constructions. This is in some sense not surprising: affine invariance is often exploited in order to prove *lower bounds* on locality [4, 5].

3.2.1 Which binomials play nice with which lines?

We would like to characterize which binomials of the form (3) restrict nicely on which lines. Unlike the case with monomials, now this will depend on the line as well as on the binomial. When both individual terms in the binomial are good monomials, the binomial will certainly restrict nicely. However, if this is not the case, then the binomial could still restrict nicely, if the contributions to the leading coefficient of $P|_L$ from the two terms cancel with each other. Using Theorem 13, we may write down these contributions and characterize when the cancel; we omit the details due to space constraints, but (see [9]) this approach can establish the following Corollary.

▶ **Corollary 16.** *Let s and t divide $q-1$, and let $G_s = \{x \in \mathbb{F}_q : x^s = 1\}$ and $G_t = \{x \in F_q : x^t = 1\}$. Let*

$$\mathcal{L}_{s,t} = \{(T, \alpha T + \beta) : \alpha \in G_s, \beta \in G_t\}$$

as in Definition 11. Suppose that $P(X,Y) = X^{a_1}Y^{b_1} + X^{a_2}Y^{b_2}$ is a binomial so that neither term is good. Suppose that $a_1 \equiv a_2 \mod s$ and $a_1 + b_1 \equiv a_2 + b_2 \mod t$. Then for all $L \in \mathcal{L}_{s,t}$, P restricts nicely to L.

Thus, a choice of s and t dividing $q-1$ produces a code by using $\mathcal{L}_{s,t}$ in Construction 1. Each choice of s and t produces a different code, and by varying s and t we can vary the parameters of this code. This is the general framework for our construction, but we still must explore the dimension and the number of disjoint repair groups produced by different choices of s and t.

3.2.2 Dimension

Given some choice of s and t, we would like to understand dimension of the space of polynomials $\mathcal{F}_{s,t}$ which restrict nicely on all lines in $\mathcal{L}_{s,t}$. We will lower bound this dimension by building a linearly independent set $S \subseteq \mathcal{F}_{s,t}$ comprised of monomials and binomials. In order to construct S and understand its size, we will need some more notation.

Let $i < s$ and $j < t$ be nonnegative integers. Define

$$E_{i,j} = \{(m,n) \in [q]^2 : m \equiv_s i, n \equiv_t j, n \leq_2 m\}.$$

Thus, the term $e(s,t)$ from Theorem 12 is the number of (i,j) so that $E_{i,j}$ is not empty. It turns out, that $E_{i,j}$ is (up to a ±1 term that we are careful about in the full version) in bijection with the set $\hat{M}_{i,j} = \left\{ X^aY^b \text{ not good } : a \equiv_s i, b + a \equiv_t j \right\}$. Notice that the sum of two monomials in $\hat{M}_{i,j}$ meets the hypotheses of Corollary 16.

This observation is at the heart of the proof of Theorem 12. In slightly more detail, we want to establish a lower bound on the dimension of polynomials which restrict nicely; to do this we will exhibit a large linearly independent set of such polynomials. We will start with all of the good monomials, and add to them a collection of binomials that satisfy Corollary 16. We can do this as follows. First, from each $\hat{M}_{i,j}$, we fix one monomial, call it $X^{a^*}Y^{b^*}$. Then, we include into our large linearly independent set all the binomails of the form $X^{a^*}Y^{b^*} + X^aY^b$ for $X^aY^b \in \hat{M}_{i,j} \setminus X^{a^*}Y^{b^*}$. Doing this for all i, j results in a collection of linearly independent binomials of size at least (ignoring some details about ±1 terms)

$$\sum_{|E_{i,j} \neq 0} (|E_{i,j} - 1) - 1 = \left(\sum_{|E_{i,j}| \neq 0} |E_{i,j}| - 1 \right) - e(s,t).$$

However, the first term, which is equal to $\sum_{i,j} |E_{i,j}| - 1$, is exactly the number of not-good monomials. So our count of good monomials, plus these binomials that restrict nicely, is precisely equal to the number of all monomials, minus $e(s,t)$. This establishes Theorem 12; we refer the reader to [9] for more details.

This theorem does give us a lower bound on the dimension of the code, but the expression depends on $e(s,t)$. We would like to know that $e(s,t)$ is not too big. It is easy to see that $e(s,t) \leq st$, because there are only st choices for (i,j). Moreover, we know that $e(s,t) \leq q^2 - g = 3^\ell - 1$, the total number of not-good monomials. As we will see in Section 4, this first bound $e(s,t) \leq st$ is nontrivial, and can in fact recover the result of $N - K = s\sqrt{N}$ of [8]. However, the point of all this work is that in fact we will be able to choose s and t so that we can get a much tighter bound on $e(s,t)$, establishing Theorem 9.

4 Instantiations

Finally, we choose t and s. One of the simplest choices we can make within our framework is to set $t = q-1$, while $s|q-1$ is any divisor. That is, we consider all simple lines $L(T) = (T, \alpha T + \beta)$ where β may vary over all of $\mathbb{F}_q^*$, and where $\alpha \in G_s$ lives in a multiplicative subgroup of $\mathbb{F}_q^*$. One reason that this choice is convenient is that it is easy to understand the number of disjoint repair groups: there are $s - 1$ lines of $\mathcal{L}_{s,q-1}$ through any nonzero point.

Thus, Theorem 12, along with the observation of the previous section that $e(s, q-1) \leq s(q-1)$ trivially, immediately implies DRGP codes that match the results of [8], with dimension $K \geq N - O(s\sqrt{N})$. However, by choosing s carefully we can actually get a tighter bound on $e(s,t)$:

▶ **Theorem 17.** *Let $q = 2^\ell$ be an even power of 2. Then*

$$e(\sqrt{q} - 1, q-1) = O\left((5 + \sqrt{5})^{\ell/2} \right).$$

Theorem 9 follows straightforwardly from Theorem 17 and Theorem 12. We omit the proof of Theorem 17 here, and refer the reader to the full version [9] for details.

5 Conclusion

We have studied the s-DRGP for intermediate values of s. As s grows, the study of the s-DRGP interpolates between the study of LRCs and LCCs, and our hope is that by understanding intermediate s, we will improve our understanding on either end of this spectrum. Using a new construction that we term a "partially lifted code," we showed how to obtain codes of length N with the s-DRGP for $s = \Theta(N^{1/4})$, that have dimension $K \geq N - N^{.714}$. This is an improvement over previous results of $N - N^{3/4}$ in this parameter regime. We stress that the main point of interest of this result is not the exponent 0.714, which we do not believe is tight for Question 1; rather, we think that our results are interesting because (a) they show that one can in fact beat $N - O(s\sqrt{N})$ for $s = N^{1/4} \ll \sqrt{N}$, and (b) they highlight the class of partially lifted codes, which we hope will be of independent interest.

Acknowledgements. We thank Alex Vardy and Eitan Yaakobi for helpful exchanges. We also thank the anonymous reviewers for suggestions which improved the paper.

References

1 Hilal Asi and Eitan Yaakobi. Nearly optimal constructions of PIR and batch codes. *CoRR*, abs/1701.07206, 2017. URL: `http://arxiv.org/abs/1701.07206`.

2 E. F. Assmus and J. D. Key. Polynomial codes and finite geometries. *Handbook of coding theory*, 2(part 2):1269–1343, 1998.

3 Megasthenis Asteris and Alexandros G. Dimakis. Repairable fountain codes. *IEEE Journal on Selected Areas in Communications*, 32(5):1037–1047, 2014.

4 Eli Ben-Sasson and Madhu Sudan. Limits on the rate of locally testable affine-invariant codes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 412–423. Springer, 2011.

5 Arnab Bhattacharyya and Sivakanth Gopi. Lower bounds for constant query affine-invariant LCCs and LTCs. In *Proceedings of the 31st Conference on Computational Complexity*, volume 50 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. `doi:10.4230/LIPIcs.CCC.2016.12`.

6 S. Blackburn and T. Etzion. PIR Array Codes with Optimal PIR Rate. *CoRR*, abs/1607.00235, 2016. URL: `http://arxiv.org/abs/1607.00235`.

7 Alexandros G Dimakis, Anna Gál, Ankit Singh Rawat, and Zhao Song. Batch codes through dense graphs without short cycles. *arXiv preprint arXiv:1410.2920*, 2014.

8 Arman Fazeli, Alexander Vardy, and Eitan Yaakobi. Codes for distributed PIR with low storage overhead. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 2852–2856. IEEE, 2015.

9 S Luna Frank-Fischer, Venkatesan Guruswami, and Mary Wootters. Locality via partially lifted codes. *arXiv preprint arXiv:1704.08627*, 2017.

10 Parikshit Gopalan, Cheng Huang, Huseyin Simitci, and Sergey Yekhanin. On the locality of codeword symbols. *IEEE Transactions on Information Theory*, 58(11):6925–6934, 2012.

11 Alan Guo and Swastik Kopparty. List-decoding algorithms for lifted codes. *CoRR*, abs/1412.0305, 2014. URL: `http://arxiv.org/abs/1412.0305`.

12 Alan Guo, Swastik Kopparty, and Madhu Sudan. New affine-invariant codes from lifting. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, ITCS'13, pages 529–540, New York, NY, USA, 2013. ACM. URL: `http://arxiv.org/abs/1208.5413`, `arXiv:1208.5413`, `doi:10.1145/2422436.2422494`.

13 Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. Local Correctability of Expander Codes. In *ICALP*, LNCS. Springer, April 2013. `arXiv:1304.8129`.

14 Cheng Huang, Minghua Chen, and Jin Li. Pyramid codes: Flexible schemes to trade space for access efficiency in reliable data storage systems. *ACM Transactions on Storage (TOS)*, 9(1):3, 2013.

15 Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Batch codes and their applications. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 262–271. ACM, 2004.

16 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *STOC'00: Proceedings of the 32nd Annual Symposium on the Theory of Computing*, pages 80–86, 2000.

17 S. Kopparty, S. Saraf, and S. Yekhanin. High-rate codes with sublinear-time decoding. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 167–176. ACM, 2011.

18 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally-correctable and locally-testable codes with sub-polynomial query complexity. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 202–215. ACM, 2016.

19 Shu Lin and Daniel J Costello. *Error control coding*. Pearson Education India, 2004.

20 Sankeerth Rao and Alexander Vardy. Lower bound on the redundancy of PIR codes. *CoRR*, abs/1605.01869, 2016. URL: `http://arxiv.org/abs/1605.01869`.

21 Ankit Singh Rawat, Dimitris S. Papailiopoulos, Alexandros G. Dimakis, and Sriram Vishwanath. Locality and availability in distributed storage. In *2014 IEEE International Symposium on Information Theory*, pages 681–685. IEEE, 2014.

22 I. Reed. A class of multiple-error-correcting codes and the decoding scheme. *Information Theory, Transactions of the IRE Professional Group on*, 4(4):38–49, September 1954.

23 Maheswaran Sathiamoorthy, Megasthenis Asteris, Dimitris Papailiopoulos, Alexandros G Dimakis, Ramkumar Vadali, Scott Chen, and Dhruba Borthakur. Xoring elephants: Novel erasure codes for big data. In *Proceedings of the VLDB Endowment*, volume 6, pages 325–336. VLDB Endowment, 2013.

24 Vitaly Skachek. Batch and PIR codes and their connections to locally-repairable codes. *CoRR*, abs/1611.09914, 2016. URL: `http://arxiv.org/abs/1611.09914`.

25 Itzhak Tamo and Alexander Barg. Bounds on locally recoverable codes with multiple recovering sets. In *2014 IEEE International Symposium on Information Theory*, pages 691–695. IEEE, 2014.

26 Itzhak Tamo and Alexander Barg. A family of optimal locally recoverable codes. *IEEE Transactions on Information Theory*, 60(8):4661–4676, 2014.

27 Itzhak Tamo, Alexander Barg, and Alexey Frolov. Bounds on the parameters of locally recoverable codes. *IEEE Transactions on Information Theory*, 62(6):3070–3083, 2016.

28 Anyu Wang and Zhifang Zhang. Repair locality with multiple erasure tolerance. *IEEE Transactions on Information Theory*, 60(11):6979–6987, 2014.

29 David P. Woodruff. *A Quadratic Lower Bound for Three-Query Linear Locally Decodable Codes over Any Field*, pages 766–779. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

30 Sergey Yekhanin. Locally Decodable Codes. *Foundations and Trends in Theoretical Computer Science*, 2010.

Testing Hereditary Properties of Sequences

Cody R. Freitag[1], Eric Price[2], and William J. Swartworth[3]

1 Department of Computer Science, UT Austin, Austin, TX, USA
cody@rdfriday.com
2 Department of Computer Science, UT Austin, Austin, TX, USA
ecprice@cs.utexas.edu
3 Department of Computer Science, UT Austin, Austin, TX, USA
wswartworth@gmail.com

Abstract

A hereditary property of a sequence is one that is preserved when restricting to subsequences. We show that there exist hereditary properties of sequences that cannot be tested with sublinear queries, resolving an open question posed by Newman et al. [20]. This proof relies crucially on an infinite alphabet, however; for finite alphabets, we observe that any hereditary property can be tested with a constant number of queries.

1998 ACM Subject Classification F.2 Analysis of Algorithms and Problem Complexity

Keywords and phrases Property Testing

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.44

1 Introduction

Property testing is the problem of distinguishing objects x that satisfy a given property P from ones that are “far” from satisfying it in some distance measure [13], with constant (say, 2/3) success probability. The most basic questions in property testing are which properties can be tested with constant queries; which properties cannot be tested without reading almost the entire input x; and which properties lie in between.

This paper considers property testing of sequences under the edit distance. We say a length n sequence x is ϵ-far from another (not necessarily length-n) sequence y if the edit distance is at least ϵn. One of the key problems in property testing is testing if a sequence is monotone; a long line of work (see [10, 5, 7, 8] and references therein) showed that $\Theta(\frac{1}{\epsilon}\log n)$ queries are necessary and sufficient.

One can generalize monotonicity by considering properties defined by forbidden order patterns. For instance, avoiding the $(1, 3, 2)$ pattern would mean that x contains no length-3 subsequence with the first smaller than the third element and the third element smaller than the second. Monotonicity would correspond to avoiding the $(2, 1)$ sequence. Pattern free sequences have a long history of study in combinatorics, such as the (now proven) Stanley-Wilf conjecture [19, 12]. In property testing, Newman et al. recently showed (among other results) that every length-k pattern can be tested with $O(n^{1-1/k}/\epsilon^{1/k})$ nonadaptive queries [20], and that $\Omega(n^{1-2/(k+1)})$ queries are necessary for testers that make non-adaptive queries.

Properties defined by forbidden order patterns can be further generalized to hereditary properties of sequences. We say a sequence property P is *hereditary* if, for any sequence x satisfying P, any subsequence of x also satisfies P. Newman et al. [20] pose as an open problem the question we consider in this work: *can any hereditary property of sequences be tested with sublinear query complexity*?

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 44; pp. 44:1–44:10
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Hereditary properties have long been studied for graphs. It was shown by [2] that hereditary properties of dense graphs are essentially precisely the ones that are testable with a constant number of queries. Similar results have been shown for hypergraphs [3] and certain sparse graphs [9].

Hereditary properties are also testable for permutations, under multiple notions of distance measure [16, 4, 17]. Since hereditary properties on graphs and permutations are testable, might they also be testable on sequences? For sequences the query complexity cannot be independent of n, since (for example) monotonicity testing requires $\Omega(\frac{1}{\epsilon}\log n)$ queries, but one could hope for something sublinear.

Our results. Our main result is to resolve the open question in the negative: there exist hereditary properties of sequences that cannot be tested with sublinear queries. We show how to reduce an arbitrary sequence property to a hereditary property over a larger alphabet. Since there exist sequence properties that require $\Omega(n)$ queries for constant ϵ, the same must hold true for hereditary properties:

▶ **Theorem 1.** *Let $\epsilon \leq 1/40$. There exist hereditary properties of sequences for which no ϵ-tester with two-sided error exists that uses $o(n)$ queries.*

Our reduction makes the sequence alphabet grow with n. While large alphabets often makes sense for sequence testing problems – for instance, forbidden order patterns typically expect all n sequence elements to be distinct – one may wonder if hereditary properties over finite alphabets behave differently. They do. We show that every hereditary property of sequences over a finite alphabet can be tested with a constant number of queries:

▶ **Theorem 2.** *Every hereditary property over a finite alphabet is testable with query complexity independent of n.*

Related work. A recent concurrent work [1] studies hereditary properties of edge-colored vertex-ordered graphs. They show that any hereditary property, for a fixed finite alphabet of edge colors, is testable with a constant number of queries. This is analogous to our upper bound for finite alphabets, but in the setting of ordered dense graphs rather than sequences.

Our Theorem 1 relies on finding a property that requires $\Omega(n)$ queries. The existence of such a property was shown in [6] for quantum property testers under Hamming distance, building on techniques in [14]. These techniques could be converted into our setting of classical property testers under edit distance. Instead, we choose to give an *explicit* property requiring $\Omega(n)$ queries for our setting, which may be of independent interest.

1.1 Overview of Techniques

This paper consists of three technical pieces: a reduction from arbitrary properties to hereditary properties over a larger alphabet; a lower bound for arbitrary properties; and an upper bound for hereditary properties over finite alphabets. We briefly outline each part in turn.

The reduction. In Section 3, we give a reduction showing that given a blackbox tester for hereditary properties using $q(n, \epsilon)$ queries, we can test arbitrary properties with $q(n, \epsilon/2)$ queries. The key to this transformation is making new, disjoint alphabets for each sequence length for the original property. Then, we can make that property hereditary by adding all subsequences. Because all alphabets are disjoint, the fact that the new property is hereditary doesn't make the property much easier to test.

Explicit hard properties. We construct an explicit property P of integer sequences which requires linear queries to test. Our construction consists of sequences over $\mathbb{F}_p$ where p grows linearly with the length of the sequence. We construct P such that a random sequence in P of length n is indistinguishable (in the information-theoretic sense) from a uniformly random sequence over $\mathbb{F}_p$ to any algorithm making fewer than $n/2$ queries. By making our property small enough, we ensure that almost all sequences over $\mathbb{F}_p$ of length n are ϵ-far from P. Thus we show that a correct tester would be able to distinguish a uniform sample from P from a uniform sample over the total space with good probability. Since this requires $n/2$ queries, we obtain a linear lower bound for testing P.

Finite-alphabet hereditary properties are easy. In Section 4, we show that testing for a hereditary property over a finite alphabet is equivalent to testing for the avoidance of a finite set of forbidden subsequences. If a sequence is ϵ-far from avoiding m subsequences under edit distance, then it must be at least ϵ/m-far from avoiding one such subsequence. This subsequence has some finite length k, which we show means that a uniform sample of $O(\frac{m}{\epsilon}k^2 \log k)$ indices finds this subsequence with constant probability.

2 Notation

A sequence of length n over an alphabet Σ is a function $S\colon [n] \to \Sigma$, often written as $(S_1, \dots, S_n)$. A property P is a set of sequences, and we say a particular sequence S has property P if S is in P. We say that a sequence S of length n is ϵ-far from P if for all $x \in P$, $d(S, x) > \epsilon n$ for some distance measure d. In this paper we consider edit distance, i.e., $d(x, y)$ is the minimum number of symbol deletions, insertions, or substitutions needed to transform x into y.

A property P is *hereditary* if for all sequences S in P, every subsequence of S is also in P. For every property P, there is a smallest hereditary property containing P, which consists of all subsequences of elements in P. We call this property the *hereditary closure* of P and denote it by P^*.

An ϵ-tester for a property P is a randomized algorithm that on an input sequence S queries a set of indices of S (possibly adaptively) and accepts with probability at least $2/3$ if $S \in P$ and rejects with probability at least $2/3$ if S is ϵ-far from P. Such a tester is said to have *two-sided error*. If the tester is instead required to accept with probability 1 on all inputs in P, we say that the tester has *one-sided error*. We say that a property P is *testable* with $q(n, \epsilon)$ queries if for every $\epsilon > 0$ there is an ϵ-tester for P using at most $q(n, \epsilon)$ queries on sequences of length n with two-sided error.

3 Hereditary Properties over Arbitrary Alphabets

Our goal in this section is to prove Theorem 1:

▶ **Theorem 1.** *Let $\epsilon \leq 1/40$. There exist hereditary properties of sequences for which no ϵ-tester with two-sided error exists that uses $o(n)$ queries.*

We first give a reduction from arbitrary property testing on sequences to hereditary property testing. The result then follows from the existence of sequence properties that cannot be tested with sublinear queries.

3.1 Reduction from Testing Arbitrary Properties to Hereditary Properties

▶ **Lemma 3.** *Fix an arbitrary infinite alphabet Σ. If every hereditary property of sequences over Σ is testable with $q(n, \epsilon)$ queries, then every property of sequences over Σ is testable with $q(n, \epsilon/2)$ queries.*

Proof. Let P be an arbitrary property over the alphabet Σ. Since Σ is infinite, there is a countably infinite collection,$\{\Sigma_1, \Sigma_2, \ldots\}$, of disjoint subsets of Σ where each Σ_m has the same cardinality as Σ [1]. For each m, let $f_m : \Sigma \to \Sigma_m$ be a fixed bijection from Σ to Σ_m.

We construct a property Q by converting every sequence in P of length m to the corresponding alphabet Σ_m. More formally, let $Q_m = \{f_m(S) \mid S \in P, S \text{ is of length } m\}$ for each $m \in \mathbb{N}$, and let $Q = \bigcup_{m\in\mathbb{N}} Q_m$.

We claim that if S is in P, then $f_m(S)$ is in the hereditary closure Q^* of Q, and if S is ϵ-far from P, then $f_m(S)$ is $\epsilon/2$-far from Q^*. It will follow from this that an $\epsilon/2$ tester for the hereditary property Q^* suffices to test for P.

Suppose S is length n and has property P. Then $f_n(S) \in Q \subseteq Q^*$, so $f_n(S)$ is in Q^*. Now suppose that S is ϵ-far from P. Trivially $f_n(S)$ is ϵ-far from every subsequence of a sequence in Q_i^* with $i \neq n$ since Σ_i and Σ_n are disjoint. Also, $f_n(S)$ is ϵ-far from every sequence in Q_n since f_n is a bijection between Σ and Σ_n. If $f_n(S)$ were $\epsilon/2$-close to a subsequence x' of some $x \in Q_n$, then x' must have length at least $n - \epsilon n/2$. This means x' is $\epsilon/2$-close to x in edit distance. It then follows that $f_n(S)$ is ϵ-close to $x \in Q_n$, which is a contradiction. Therefore, $f_n(S)$ must be $\epsilon/2$-far from Q^*. ◀

3.2 An Explicit Property Requiring Linear Queries

Related work uses a nonconstructive argument to show that there exists properties of binary sequences which require linear queries to test with two-sided error [6]. Here we construct an explicit class of sequences over $\mathbb{Z}$ which require linear queries. Specifically we show that testing whether a vector in $\mathbb{F}_p^{2n}$ lies in the space of codewords of a Reed-Solomon code requires at least n queries.

For $p \geq k$, let Reed-Solomon$_p(l, k)$ denote the space of codewords for the Reed-Solomon code over $\mathbb{F}_p$ with message length l and codeword length k. Explicitly we define Reed-Solomon$_p(l, k)$ to be the column span of the following matrix taken over $\mathbb{F}_p$:

$$\begin{bmatrix} 1^0 & 1^1 & \ldots & 1^{l-1} \\ 2^0 & 2^1 & \ldots & 2^{l-1} \\ 3^0 & 3^1 & \ldots & 3^{l-1} \\ \vdots & \vdots & \ddots & \vdots \\ k^0 & k^1 & \ldots & k^{l-1} \end{bmatrix}.$$

Our main result is that when k is larger than l by a constant factor, testing for membership in Reed-Solomon$_p(l, k)$ requires linear queries.

▶ **Lemma 4.** *Let P be the space of codewords for Reed-Solomon$_p(n, 2n)$, and set $\epsilon = 1/40$. An adaptive two sided tester (with $2/3$ success probability), which ϵ-tests for P must make at least n queries.*

[1] For arbitrary Σ, this result requires the axiom of choice. However in the case $\Sigma = \mathbb{N}$ we may be explicit by setting $\Sigma_m = \{(m + i)^2 + i | i \in \mathbb{N}\}$.

We require the following well-known property of the Reed-Solomon matrix M.

▶ **Lemma 5.** *Let M be the $2n \times n$ matrix with $M_{i,j} = i^{j-1}$. Each $n \times n$ submatrix of M has full rank.*

Proof. Let $v = [v_0, \ldots v_{n-1}]^T$, and let M_i denote the i^{th} row of M. Set

$$q_v(x) = v_0 + v_1 x^1 + \ldots + v_{n-1} x^{n-1},$$

and observe that that $M_i v = q_v(i)$. If some n rows of M were dependent then for some nonzero v we would have $M_i v = q_v(i) = 0$ for n different values of i. But this cannot happen since q_v is a nonzero polynomial of degree at most $n-1$. ◀

Our main argument proceeds by showing that a tester for P would be able distinguish a sequence drawn from the uniform distribution on P from a sequence drawn from the uniform distribution on $\mathbb{F}_p^{2n}$ with good probability. We will first argue this fact, and then show that any algorithm which distinguishes these distribution with probability greater than $1/2$ must make at least n queries.

The first step amounts to bounding the size of an ϵ-ball in $\mathbb{F}_p^{2n}$.

▶ **Lemma 6.** *The size of an ϵ-ball in F_p^n under edit distance is at most $(ep/\epsilon)^{2\epsilon n}$.*

Proof. Recall that under our definitions, edit distance allows for insertions, deletions, and replacements. A replacement may be simulated with a deletion, followed by an insertion. Therefore, if $d(\cdot,\cdot)$ is the analogue of edit distance allowing only insertions and deletions as moves, it suffices to bound the size of a 2ϵ-ball under the metric d.

Fix $x \in F_p^n$. Any element in $B_d(2\epsilon, x)$ may be constructed from x by the following procedure. First we select a subset of ϵn indices of x to delete. Then we choose a multiset of indices in $\{0, 1, \ldots n - \epsilon n\}$ of size ϵn corresponding to the locations in the resulting sequence where we will perform our insertions. Finally we choose a sequence of length ϵn to insert into those locations.

There are $\binom{n}{\epsilon n}$ ways to choose the ϵn elements to delete. Then there are $\binom{(n-n\epsilon)+n\epsilon}{n\epsilon} = \binom{n}{n\epsilon}$ ways to select the multiset of indices of size ϵn. Finally there are $p^{\epsilon n}$ ways to choose a sequence of length ϵn. It follows that

$$\begin{aligned}|B_d(2\epsilon, x)| &\leq \binom{n}{n\epsilon} \cdot \binom{n}{n\epsilon} \cdot p^{\epsilon n}\\ &\leq \left(\frac{e}{\epsilon}\right)^{2\epsilon n} \cdot p^{\epsilon n}\\ &\leq \left(\frac{ep}{\epsilon}\right)^{2\epsilon n}.\end{aligned}$$ ◀

▶ **Lemma 7.** *Set $\epsilon = 1/40$, and let T be an ϵ-tester for P. For $x \sim \mathit{Uniform}(\mathbb{F}_p^{2n})$, T will accept with probability strictly less than $1/2$ (for large enough n).*

Proof. The argument is that a uniformly random vector in $\mathbb{F}_p^{2n}$ is ϵ-far from P (in edit distance) with high probability. We first observe that an ϵ-neighborhood of P is small. In particular we have

$$\begin{aligned}|\{x \in \mathbb{F}_p^{2n} : x \text{ is } \epsilon\text{-close to } P\}| &\leq |B_\epsilon| \cdot |P|\\ &\leq \left(\frac{ep}{\epsilon}\right)^{4\epsilon n} \cdot p^{2n/2}\\ &\leq (60p)^{n/10} \cdot p^n\\ &\leq p^{7n/10} \cdot p^n\\ &\leq p^{1.7n},\end{aligned}$$

where we used that $p \geq 2$.

The probability that a vector drawn uniformly from $\mathbb{F}_p^{2n}$ is ϵ-close to P is at most $p^{1.7n}/p^{2n}$ which in turn is at most $2^{-0.3n}$. Therefore for $x \sim \text{Uniform}(\mathbb{F}_p^{2n})$, and $n > 6$, we have

$$\Pr[T \text{ rejects on } x] \geq (2/3) \cdot (1 - 2^{-0.3n}) > 1/2,$$

since T must reject, with probability 2/3, every point which is ϵ-far from P. ◀

The next step is to argue that any tester which makes fewer than n queries, cannot distinguish the distributions $\text{Uniform}(\mathbb{F}_p^{2n})$ and $\text{Uniform}(P)$. In fact we have the following:

▶ **Lemma 8.** *Let x and y be random vectors draw from Uniform($\mathbb{F}_p^{2n}$) and Uniform(P) respectively. For any collection $\mathcal{I} \subseteq [2n]$ of indices with $|\mathcal{I}| \leq n$, the distributions on $x|_\mathcal{I}$ and $y|_\mathcal{I}$ are both uniform over vectors of length $|\mathcal{I}|$*

Proof. It is immediately clear that $x|_\mathcal{I}$ is uniform. That $y|_\mathcal{I}$ is uniform follows from the construction of the matrix A. To be precise, first recall that the restriction of A to any collection n rows is an invertible matrix. It follows that for any $m \leq n$, the restriction of A to any m rows has rank m. The column span of a full-rank $m \times n$ matrix over $\mathbb{F}_p$ is exactly $\mathbb{F}_p^m$. Therefore $y|_\mathcal{I}$ is uniform over vectors of length $|\mathcal{I}|$. ◀

Putting these facts together completes the proof of Theorem 4.

Proof. Let x be a vector in $\mathbb{F}_p^{2n}$ sampled either from $\text{Uniform}(\mathbb{F}_p^{2n})$ or $\text{Uniform}(P)$. Suppose that our tester T makes at most n queries on x, possibly adaptively. By Lemma 8, the value at each index in x after fewer than n queries is uniformly random over $\mathbb{F}_|$ and independent of the values of all previous queries. Hence for either distribution we may simulate T's behavior by returning uniformly random values for each of its queries. Therefore T must have the same probability of acceptance on both of the two distributions for x. Lemma 7 shows that a correct T must accept on $\text{Uniform}(\mathbb{F}_p^{2n})$ with probability smaller than 1/2. But by correctness, T must accept on $\text{Uniform}(P)$ with at least 2/3 probability. It follows that a T which makes fewer than n queries cannot be correct. ◀

4 Hereditary Properties over Finite Alphabets

We now show that the reduction of Section 3.1 relied heavily on the fact the the resulting hereditary property was over an infinite alphabet. In fact, hereditary properties over a finite alphabet can be tested with sublinear query complexity.

▶ **Theorem 2.** *Every hereditary property over a finite alphabet is testable with query complexity independent of n.*

We begin with the following standard definition:

▶ **Definition 9.** A partial order $(P, \preceq)$ is said to be a *well partial order* if for every infinite sequence $p_1, p_2, \ldots$ of elements in P, there exists $i < j$ such that $p_i \preceq p_j$.

As mentioned in [18], the following result is well-known. We present a proof here mostly for completeness. A similar proof is presented in [15] but we provide a different exposition which exploits some general structural properties of well partial orders.

▶ **Lemma 10.** *Finite length sequences over a finite alphabet form a well partial order with respect to the subsequence relation.*

The proof of Lemma 10 relies on the following two lemmas.

▶ **Lemma 11.** *Let P be a well partially ordered set, and let $X = x_1, x_2, \ldots$ be a sequence of elements from P. Then there is a subsequence $Y = y_1, y_2, \ldots$ of X, such that $y_i \leq y_j$ for all $i \leq j$.*

Proof. First we argue that there exists an x_i which is (weakly) dominated by infinitely many elements of X. Suppose not. Then for each x_i, let i' be the largest integer satisfying $x_i \leq x_{i'}$. Let S denote the sequence of X corresponding to the set $\{x_{i'} : i \in \mathbb{N}\}$. Since S is necessarily infinite, there exists elements $s_i \leq s_j$ with $i < j$. But this contradicts the maximality of the $x_{i'}$'s.

To construct the sequence Y, we take y_1 to be x_{i_1}, where x_{i_1} is dominated by infinitely many elements in X. Set $S_1 = \{x_k : k > i_1, x_k \geq x_{i_1}\}$. Since S_1 is infinite, we may take y_2 to be x_{i_2} where x_{i_2} is dominated by infinitely many elements of S_1. By iterating this procedure we obtain our sequence Y. ◀

▶ **Lemma 12.** *Let $P_1, \ldots P_n$ be sets which are well partially ordered. Order the set $P_1 \times \ldots \times P_n$ by termwise domination. That is we say that $(p_1, \ldots, p_n) \leq (p'_1, \ldots p'_n)$ if and only if $p_i \leq p'_i$ for all $i \in [n]$. With this order, $P_1 \times \ldots \times P_n$ is a well partial order.*

Proof. By a straightforward induction, it suffices to prove the result when $n = 2$. Consider a sequence $S = \{(a_i, b_i)\}$ with $a_i \in P_1$ and $b_i \in P_2$. By Lemma 11 applied to P_1, there is an infinite subsequence of tuples S' such the first entries in each element of S' are (weakly) increasing. Now since P_2 is a well partial order, there exists elements $s'_i \leq s'_j$ in S' with $i < j$. Since S' is a subsequence of S it follows that S is a well partial order. ◀

Now we present a proof of Lemma 10.

Proof. Let $\mathcal{A}_k = \{a_1, \ldots, a_k\}$ be our finite alphabet of size k. Our proof is by induction on k. When $k = 1$ the result follows from $\mathbb{N}$ being a well partial order.

Now fix an alphabet of size $k+1$. Consider an infinite sequence $X = x_1, x_2, \ldots$ consisting of finite strings over the alphabet $\mathcal{A}_{k+1}$. Given a finite string $S = s_1, \ldots s_n$ over the alphabet $\mathcal{A}_{k+1}$ we represent it as a tuple $(u_1, \ldots, u_m)$ satisfying the following considerations:

- u_i is a finite sequence over the alphabet $\mathcal{A}_{k+1} - \{a_{i \mod (k+1)}\}$
- S is the concatenation of the strings $u_1, \ldots u_n$.
- each u_i is as long as possible, i.e. the first character of u_{i+1} is $a_{i \mod (k+1)}$.

Using the final property listed above, we observe that if this tuple has size at least $r(k+1)+1$, then S contains the subsequence $(a_1, a_2, \ldots, a_{k+1})^r$, where the exponent means that we repeat the string inside the parentheses r times.

Now represent each element of the sequence X as a tuple in this way. If x_1 is contained as a subsequence in some x_i with $i > 1$ then we are finished. Otherwise, let x_1 have length l. Then x_1 is contained as a substring in $(a_1 a_2 \ldots a_{k+1})^l$. The tuple associated to each x_i with $i > 1$ must have length at most $l(k+1)+1$. Otherwise, by our previous observation, x_i would contain $(a_1 a_2 \ldots a_{k+1})^l$ as a substring, and hence also x_1. We may represent each x_i with a tuple of length exactly $l(k+1)+1$ by padding x_i's tuple with empty strings as necessary. By induction, the elements of these tuples are well partially ordered. But then Lemma 12 implies that the tuples of length $l(k+1)+1$ also form a well partial order. Since the ordering on strings respects the ordering on tuples, it follows that there exists $i < j$ with $x_i \leq x_j$. Therefore X is well partially ordered. ◀

We are now ready to prove the following key fact.

▶ **Lemma 13.** *Let P be a hereditary property of sequences over a finite alphabet Σ. Then there exists a finite set $\mathcal{S}$ of sequences over Σ such that P consists exactly of the sequences which do not contain any sequence in $\mathcal{S}$ as a subsequence.*

Proof. First observe that since P is hereditary, P consists of all sequences which do not contain any sequence in $\overline{P}$, the complement of P, as a subsequence. Since $\overline{P}$ is countable, we may enumerate it as $\overline{P} = \{q_1, q_2, \ldots\}$. We construct $\mathcal{S}$ inductively, by setting $s_1 = q_1$, and setting $s_{i+1} = q_j$ where j is the minimum value such that q_j does not contain any of the sequences $s_1, \ldots s_i$ as a subsequence. Lemma 10 implies that this process must halt at some point by the definition of a well partial order, so $\mathcal{S}$ will be finite. From the construction, it is clear that each sequence in $\overline{P}$ contains a sequence in $\mathcal{S}$ as a subsequence. Therefore, P is exactly the set of sequences that avoid sequences in $\mathcal{S}$ as a subsequence. ◀

With these results, we give a short proof of Theorem 2.

Proof. By Lemma 13 it suffices to construct a tester that tests whether an input x avoids a finite collection of forbidden subsequences. In fact it is enough to construct a tester for each such sequence individually. This is because if x is ϵ-far from avoiding a collection of m sequences, then x must be ϵ/m-far from avoiding one of these subsequences. This relies on the fact that we are using edit distance, so to avoid a particular subsequence, we can just delete a subset of indices that contain that subsequence.

Suppose x were ϵ/m-close to avoiding m subsequences, $y_1, \ldots, y_m$, individually. Let S_i be the smallest set of indices such that deleting S_i from x causes x to avoid y_i. Note that by assumption of x being ϵ/m-close to avoiding y_i, $|S_i| \leq \epsilon n/m$. Then deleting $\cup_{i=1}^m S_i$ from x will cause x to avoid all m subsequences, but $| \cup_{i=1}^m S_i| \leq m \cdot (\epsilon n/m) = \epsilon n$. This contradicts that x is ϵ-far from avoiding all of $y_1, \ldots, y_m$. Therefore constructing an ϵ/m-tester for avoiding a particular sequence suffices.

Let u be a forbidden subsequence of size k. If x is ϵ-far from avoiding u, x must have at least $\epsilon n/k$ disjoint copies of u as subsequences. It was noted in [20] that a uniform sample of $O(\epsilon^{1/k} n^{1-1/k})$ entries contains one of these subsequences with constant probability by a second moment bound. However, we show in Lemma 14 that over a finite alphabet, this can be improved to just a uniform sample of $O(\frac{1}{\epsilon} k^2 \log k)$ entries.

Then to test whether x has a hereditary property over a finite alphabet, we compute the m forbidden subsequences, each of length say at most k. Then after sampling $O(\frac{m}{\epsilon} k^2 \log k)$ random indices, if x is ϵ-far from avoiding all forbidden subsequences, we will find the subsequence that x is ϵ/m-far from avoiding with at least $2/3$ probability. ◀

▶ **Lemma 14.** *There exists an ϵ-tester with one-sided error for avoiding a fixed subsequence s of length k using $O(\frac{1}{\epsilon} k^2 \log k)$ queries.*

Proof. We first assume that k is a power of 2 and then reduce to the case of general k. We also use the fact that if a sequence x is ϵ-far from avoiding s as a subsequence, then there must be a set T consisting of $\epsilon n/k$ disjoint copies of s in x [20].

Let i be minimal such that the restriction of x to T contains at least $|T|/2 = \epsilon n/2k$ disjoint instances of the subsequence $s_1, \ldots s_{k/2}$ strictly to the left of i. By minimality of i it follows that $x_i, x_{i+1}, \ldots, x_n$ contains at least $\epsilon n/2k - 1$ disjoint copies of $s_{k/2+1}, \ldots, s_k$. By iterating this procedure, we divide x into k blocks $X_1, \ldots X_k$ such that each X_i contains at least $\epsilon n/k^2 - \log k$ copies of s_i, which is $\Omega(\epsilon n/k^2)$ as long as $k = o(n^{1/2})$.

Our algorithm is to sample a uniform subset of x of size u. The probability any individual sample will be an instance of s_i from the block X_i is at least $\Omega(\epsilon/k^2)$. Thus with constant probability, we will select a corresponding s_i from each of the blocks X_i after $O(\frac{1}{\epsilon}k^2 \log k)$ samples.

We now reduce the case where the length of the subsequence is a power of 2 to general k. Let s be of length k, and k' be the smallest power of 2 larger than k. Let c be any character not in the alphabet of the sequence. We will construct s' of length k' by adding $k' - k$ copies of c to the end of s. We also construct the sequence x' by adding $(k' - k) \cdot \epsilon n/k$ copies of c to the end of x.

Note that x' avoids s' if and only if x avoids s since c is disjoint from the original alphabet. Also $k' - k < k$, so the length of x' is at most $2n$. This means x is ϵ-far from avoiding s if and only if x' is at least $\epsilon/2$-far from avoiding s'. Also, we can simulate any property testing algorithm on x' since any query for an index greater than n must return c. Therefore we can test x for s-avoidance by testing x' for s'-avoidance using $O(\frac{1}{\epsilon/2}(k')^2 \log k') = O(\frac{1}{\epsilon}k^2 \log k)$ queries. ◀

5 Conclusions and Open Problems

We showed that there exist hereditary properties that require linear query complexity. However, we also show that when we restrict to hereditary properties over a finite alphabet, there are testers using queries independent of n. What can we say about other natural restrictions on hereditary properties? Sequences over an infinite alphabet don't form a well-partial order under the subsequence relation, as shown in [21], so we need different techniques to see if other interesting restrictions over infinite alphabets can be tested using sublinear queries.

One natural restriction is to order-based hereditary properties [11]. [20] considers testing the avoidance of permutation patterns, which is a subclass of order-based hereditary properties. A sequence S avoids a pattern π of length k if there is no set of indices $i_1 < i_2 < \ldots < i_k$ such that $S_{i_x} > S_{i_y}$ if and only if $\pi_x > \pi_y$. It is unknown whether testing the avoidance of constant length patterns requires more than polylog(n) queries with adaptive algorithms.

References

1 Noga Alon, Omri Ben-Eliezer, and Eldar Fischer. Testing hereditary properties of ordered graphs and matrices. *arXiv preprint arXiv:1704.02367*, 2017.

2 Noga Alon and Asaf Shapira. A characterization of the (natural) graph properties testable with one-sided error. *SIAM Journal on Computing*, 37(6):1703–1727, 2008.

3 Tim Austin and Terence Tao. Testability and repair of hereditary hypergraph properties. *Random Structures & Algorithms*, 36(4):373–463, 2010.

4 Antônio J. O. Bastos, Carlos Hoppen, Yoshiharu Kohayakawa, and Rudini M. Sampaio. Every hereditary permutation property is testable. *Electronic Notes in Discrete Mathematics*, 38:123–128, 2011.

5 Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. *SIAM Journal on Computing*, 41(6):1380–1425, 2012.

6 Harry Buhrman, Lance Fortnow, Ilan Newman, and Hein Röhrig. Quantum property testing. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 480–488. Society for Industrial and Applied Mathematics, 2003.

7 Deeparnab Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and lipschitz testing over hypercubes and hypergrids. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 419–428. ACM, 2013.

8 Deeparnab Chakrabarty and C. Seshadhri. An optimal lower bound for monotonicity testing over hypergrids. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 425–435. Springer, 2013.

9 Artur Czumaj, Asaf Shapira, and Christian Sohler. Testing hereditary properties of non-expanding bounded-degree graphs. *SIAM Journal on Computing*, 38(6):2499–2510, 2009.

10 Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *Randomization, Approximation, and Combinatorial Optimization. Algorithms and Techniques*, pages 97–108. Springer, 1999.

11 Eldar Fischer. On the strength of comparisons in property testing. *Information and Computation*, 189(1):107–116, 2004.

12 Jacob Fox. Stanley-wilf limits are typically exponential. *arXiv preprint arXiv:1310.8378*, 2013.

13 Oded Goldreich. Combinatorial property testing (a survey). *Randomization Methods in Algorithm Design*, 43:45–59, 1999.

14 Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998.

15 Leonard H. Haines. On free monoids partially ordered by embedding. *Journal of Combinatorial Theory*, 6(1):94–98, 1969.

16 Carlos Hoppen, Yoshiharu Kohayakawa, Carlos Gustavo Moreira, and Rudini Menezes Sampaio. Testing permutation properties through subpermutations. *Theoretical Computer Science*, 412(29):3555–3567, 2011.

17 Tereza Klimošová and Daniel Král. Hereditary properties of permutations are strongly testable. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1164–1173. Society for Industrial and Applied Mathematics, 2014.

18 Joseph B. Kruskal. The theory of well-quasi-ordering: A frequently discovered concept. *Journal of Combinatorial Theory, Series A*, 13(3):297–305, 1972.

19 Adam Marcus and Gábor Tardos. Excluded permutation matrices and the stanley–wilf conjecture. *Journal of Combinatorial Theory, Series A*, 107(1):153–160, 2004.

20 Ilan Newman, Yuri Rabinovich, Deepak Rajendraprasad, and Christian Sohler. Testing for forbidden order patterns in an array. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1582–1597. SIAM, 2017.

21 Daniel A Spielman and Miklós Bóna. An infinite antichain of permutations. *Electron. J. Combin*, 7:N2, 2000.

Traveling in Randomly Embedded Random Graphs

Alan Frieze[1] **and Wesley Pegden**[2]

1 Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA, USA
2 Department of Mathematical Sciences, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

We consider the problem of traveling among random points in Euclidean space, when only a random fraction of the pairs are joined by traversable connections. In particular, we show a threshold for a pair of points to be connected by a geodesic of length arbitrarily close to their Euclidean distance, and analyze the minimum length Traveling Salesperson Tour, extending the Beardwood-Halton-Hammersley theorem to this setting.

1998 ACM Subject Classification G.2.1 Combinatorics, G.3 Probability and Statistics

Keywords and phrases Traveling Salesman, Euclidean, Shortest Path

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.45

1 Introduction

The classical Beardwood-Halton-Hammersley theorem [1] (see also Steele [12] and Yukich [13]) concerns the minimum cost Traveling Salesperson Tour through n random points in Euclidean space. In particular, it guarantees the existence of an absolute (though still unknown) constant β_d such that if $X_1, X_2 \dots,$ is a random sequence of points, uniformly distributed in the d-dimensional cube $[0,1]^d$, the length $T(\mathcal{X}_{n,1})$ of a minimum tour through $X_1, \dots, X_n$ satisfies

$$T(\mathcal{X}_{n,1}) \sim \beta_d n^{\frac{d-1}{d}} \; a.s. \tag{1}$$

The present paper is concerned still with the problem of traveling among random points in Euclidean space. In our case, however, we suppose that only a (random) subset of the pairs of points are joined by traversable connections, independent of the geometry of the point set.

In particular, we study random embeddings of the Erdős-Rényi-Gilbert random graph $G_{n,p}$ into the d-dimensional cube $[0,1]^d$. We let $\mathcal{X}_n$ denote a uniformly random set of points $X_1, X_2, \dots, X_n \in [0,1]^d$, and we denote by $\mathcal{X}_{n,p}$ the random graph whose vertex set is $\mathcal{X}_n$ and whose pairs of vertices are joined by edges each with independent probability p. Edges are weighted by the Euclidean distance between their points, and we are interested in the total edge-weight required to travel about the graph.

This model has received much less attention than the standard model of a random geometric graph, defined as the intersection graph of unit balls with random centers $X_i, i \in [n]$, see Penrose [9]. We are only aware of the papers by Mehrabian [7] and Mehrabian and Wormald [8] who studied the *stretch factor* of $\mathcal{X}_{n,p}$. In particular, let $||x-y||$ denote the Euclidean distance between vertices x, y, and $\text{dist}(x,y)$ denote their distance in $\mathcal{X}_{n,p}$. They showed (considering the case $d = 2$) that unless p is close to 1, the stretch factor

$$\sup_{x,y \in \mathcal{X}_{n,p}} \frac{\text{dist}(x,y)}{||x-y||}$$

tends to ∞ with n.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 45; pp. 45:1–45:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

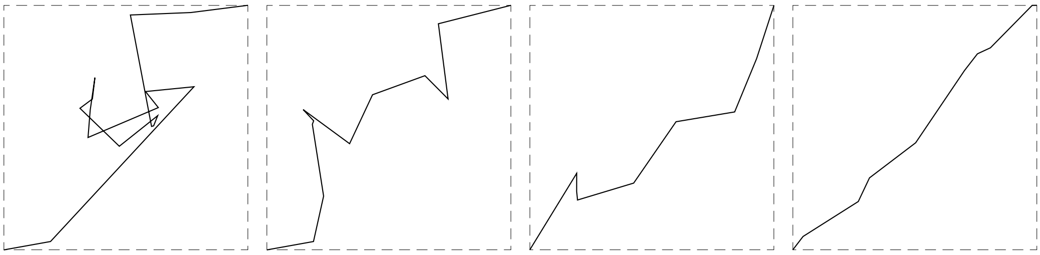


Figure 1 Paths in an instance of $\mathcal{X}_{n,p}$ for $d = 2$, $n = 2^{30}$, and $p = \frac{10}{n}, \frac{25}{n}, \frac{50}{n}$, and $\frac{200}{n}$, respectively. In each case, the path drawn is the shortest route between the vertices x and y which are closest to the SW and NE corners of the square. (See Q. 2, Section 5.)

As a counterpoint to this, our first result shows a very different phenomenon when we pay attention to additive rather than multiplicative errors. In particular, for $p \gg \frac{\log^d n}{n}$, the distance between a typical pair of vertices is arbitrarily close to their Euclidean distance, while for $p \ll \frac{\log^d n}{n(\log\log n)^{2d}}$, the distance between a typical pair of vertices in $\mathcal{X}_n$ is arbitrarily large (Figure 1). (We write $\log^k x$ for $(\log x)^k$.) In particular, this means that when $\frac{\log^d n}{n} \ll p < 1-\varepsilon$, the supremum in the stretch factor theorem of Mehrabian and Wormald is due just to pairs of vertices which are very close together.

▶ **Theorem 1.** *Let $\omega = \omega(n) \to \infty$. We have for $d \geq 2$:*

(a) *For $p \leq \frac{1}{\omega^d(\log\log n)^{2d}}\frac{\log^d n}{n}$ and fixed $u = X_1$, $v = X_2$, say, we have*

$$\mathrm{dist}(u,v) \geq \frac{\omega}{8de^d} \qquad a.a.s.^1$$

(b) *For $p \geq \frac{\omega \log^d n}{n}$, we have a.a.s. that uniformly for all pairs of vertices $u, v \in \mathcal{X}_n$,*

$$\mathrm{dist}(u,v) = ||u-v|| + o(1).$$

Theorem 1 means that, even for p quite small, it is not that much more expensive to travel from one vertex of $\mathcal{X}_{n,p}$ to another than it is to travel directly between them in the plane. On the other hand, there is a dramatic dependence on p if the goal is to travel among *all* points. Let $T(\mathcal{X}_{n,p})$ denote the length of a minimum length tour in $\mathcal{X}_{n,p}$ hitting every vertex exactly once, i.e. a Traveling Salesperson tour.

▶ **Theorem 2.** *There exists a sufficiently large constant $K > 0$ such that for all $p = p(n)$ such that $p \geq \frac{K \log n}{n}$, $d \geq 2$, we have that*

$$T(\mathcal{X}_{n,p}) = \Theta\left(\frac{n^{\frac{d-1}{d}}}{p^{1/d}}\right) \qquad a.a.s. \tag{2}$$

(Recall that $f(n) = \Theta(g(n))$ means that $f(n)$ is bounded between positive constant multiples of $g(n)$ for sufficiently large n.) As the threshold for $G_{n,p}$ to be Hamiltonian is at $p = \frac{\log n + \log\log n + \omega(n)}{n}$ (see e.g. Bollobás [2]), this theorem covers nearly the entire range of p for which a TSP tour exists a.a.s.

Finally, we extend the asymptotically tight BHH theorem [1] to the case of $\mathcal{X}_{n,p}$ for any constant p. To formulate an "almost surely" statement, we let $\mathcal{X}_{\mathcal{N},p}$ denote a random graph on a random embedding of $\mathcal{N} = \{1, 2, \ldots, \}$ into $[0,1]^d$, where each pair $\{i, j\}$ is independently present as an edge with probability p, and consider $\mathcal{X}_{n,p}$ as the restriction of $\mathcal{X}_{\mathcal{N},p}$ to the first n vertices $\{1, \ldots, n\}$.

▶ **Theorem 3.** *If $d \geq 2$ and $p > 0$ is constant, then there exists $\beta_{d,p} > 0$ such that*

$$T(\mathcal{X}_{n,p}) \sim \beta_{d,p} n^{\frac{d-1}{d}} \qquad a.s.$$

Karp's algorithm [6] for a finding an approximate tour through $\mathcal{X}_n$ extends to the case $\mathcal{X}_{n,p}$, p constant as well:

▶ **Theorem 4.** *For fixed $d \geq 2$ and p constant, then there is an algorithm that a.s. finds a tour in $\mathcal{X}_{n,p}$ of value $(1+o(1))\beta_{d,p}n^{(d-1)/d}$ in polynomial time, for all $n \in \mathcal{N}$.*

2 Traveling between pairs

2.1 Proof of Theorem 1(a)

Outline of proof

This is straightforward. We show by the first moment method that any path between u and v with "many" edges must contain a significant number of "long" edges and hence must be as long as claimed. We then show that a.a.s. there are no paths between u and v without many edges.

Proof proper

Let ν_d denote the volume of a d-dimensional unit ball; recall that ν_d is bounded ($\nu_d \leq \nu_5 < 6$ for all d).

Let an edge be *long* if its length is at least $\ell_1 = \frac{\omega(\log\log n)^2}{4e^d \log n}$. Let $\varepsilon = \frac{1}{\log\log n}$ and let $\mathcal{A}_k$ be the event that there exists a path with k edges, $k \geq k_0 = \frac{\log n}{2d \log\log n}$ from u to v that uses at most εk long edges. Then

$$\mathbf{Pr}(\exists k : \mathcal{A}_k) \leq \sum_{k \geq k_0} (k-1)! \binom{n}{k-1} p^k \binom{k}{\varepsilon k} \left(\nu_d \left(\frac{\omega(\log\log n)^2}{4e^d \log n}\right)^d\right)^{(1-\varepsilon)k} \tag{3}$$

$$\leq \sum_{k \geq k_0} n^{k-1} p^k \left(\frac{e}{\varepsilon}\right)^{\varepsilon k} \left(\nu_d \left(\frac{\omega(\log\log n)^2}{4e^d \log n}\right)^d\right)^{(1-\varepsilon)k} \tag{4}$$

$$\leq \frac{1}{n} \sum_{k \geq k_0} \left(\frac{\nu_d \log^{d\varepsilon} n}{(4e^d)^{d(1-\varepsilon)}} \cdot \left(\frac{e}{\varepsilon}\right)^{\varepsilon}\right)^k \tag{5}$$

$$\leq \frac{1}{n} \sum_{k \geq k_0} \left(\frac{6e^{d+o(1)}}{4e^{d^2}}\right)^k = o(1),$$

after using $d \geq 2$ and $\log^{\varepsilon} n = e$.

Explanation of (3): Choose the $k-1$ interior vertices of the possible path and order them in one of $(k-1)!\binom{n}{k-1}$ ways as $(u_1, u_2, \ldots, u_{k-1})$. Then p^k is the probability that the edges exist in $G_{n,p}$. Now choose the short edges $e_i = (u_{i-1}, u_i), i \in I$ in one of $\binom{k}{(1-\varepsilon)k} = \binom{k}{\varepsilon k}$ ways and bound the probability that these edges are short by $\left(\nu_d \left(\frac{\omega(\log\log n)^2}{4e^d \log n}\right)^d\right)^{(1-\varepsilon)k}$ viz. the probability that u_i is mapped to the ball of radius ℓ_1, center u_{i-1} for $i \in I$.

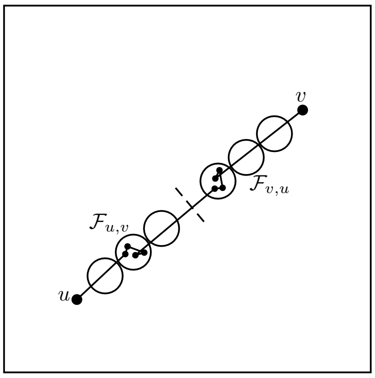


Figure 2 Finding a short path.

Now a.a.s. the shortest path in $G_{n,p}$ from u to v requires at least k_0 edges: Indeed the expected number of paths of length at most k_0 from u to v can be bounded by

$$\sum_{k=1}^{k_0}(k-1)!\binom{n}{k-1}p^k \leq \frac{1}{n}\sum_{k=1}^{k_0}\left(\frac{\log^d n}{\omega^d(\log\log n)^{2d}}\right)^k = o(1).$$

So a.a.s.

$$\mathrm{dist}(u,v) \geq \varepsilon k_0 \ell_1 = \frac{\varepsilon \log n}{2d\log\log n}\cdot\frac{\omega(\log\log n)^2}{4e^d\log n} = \frac{\omega}{8de^d}.$$

◀

2.2 Proof of Theorem 1(b)

Outline of proof

We first consider two points u, v such that $||u-v|| \geq \gamma = \frac{1}{\log\log n}$. We then consider a set of 2β small disjoint balls with centers on the line joining u, v. We argue that a.a.s. (i) all of these balls contain (relatively) giant components, (ii) there is an edge joining the large components inside each ball, (iii) the diameter of each of these giant components is small and (iv) there is an edge between u and one of the g giant components X closest to u and an edge between v and one of the g giant components Y closest to v. This gives a path consisting of an edge from u to the giant component X plus a walk inside X plus an edge to the giant component Y plus an edge to v. Because the balls are small the length of this path is close to $||u-v||$. We reduce the case where $||u-v|| \leq \gamma$ to the first case.

Proof proper

We begin by considering the case of vertices u, v at distance $||u-v|| \geq \gamma$. Letting $\delta = \frac{1}{\log n}$, then, for sufficiently large n, we can find a set $\mathcal{B}$ of at least $\frac{2C}{\delta}$, $C = \frac{\gamma}{8}$, disjoint balls of radius δ centered on the line from u to v, such that $\frac{C}{\delta}$ of the balls are closer to u than v, and $\frac{C}{\delta}$ balls are closer to v than u (Figure 2). Denote these two families of $\frac{C}{\delta}$ balls by $\mathcal{F}_{u,v}$ and $\mathcal{F}_{v,u}$. (The sets $\mathcal{B}$, $\mathcal{F}_{u,v}$ and $\mathcal{F}_{v,u}$ are fixed for the rest of the argument.)

Given a ball $B \in \mathcal{F}_{\{u,v\}} = \mathcal{F}_{u,v} \cup \mathcal{F}_{v,u}$, the induced subgraph G_B on vertices of $\mathcal{X}$ lying in B is a copy of $G_{N,p}$, where $N = N(B)$ is the (random) number of vertices lying in B. Let

$$\mathcal{S}_B \text{ be the event that } N(B) \in \left[\frac{N_0}{2^{d+1}}, 2N_0\right] \text{ where } N_0 = \nu_d\delta^d n.$$

(Dividing by 2^{d+1} accounts for points close to the boundary of $[0,1]^d$.)

Now $N(B)$ is distributed as the binomial $Bin(n,q)$ where $q \in \nu_d\delta^d[2^{-d},1]$. The following Chernoff bounds will thus be useful:

$$\mathbf{Pr}(Bin(M,p) \le (1-\varepsilon)Mp) \le e^{-\varepsilon^2 Mp/2} \text{ for } 0 \le \varepsilon \le 1. \tag{6}$$

$$\mathbf{Pr}(Bin(M,p) \ge (1+\varepsilon)Mp) \le e^{-\varepsilon^2 Mp/3} \text{ for } 0 \le \varepsilon \le 1. \tag{7}$$

The bounds (6) and (7) imply that for $B \in \mathcal{F}_{\{u,v\}}$,

$$\mathbf{Pr}\left(\neg\mathcal{S}_B\right) \le e^{-\Omega(n\delta^d)} = e^{-n^{1-o(1)}}.$$

This gives us that a.a.s. $\mathcal{S}_B$ occurs for all pairs $u,v \in \mathcal{X}$ with $||u-v|| \ge \gamma$. We now argue that for all $B \in \mathcal{B}$:

(A) All subgraphs G_B for $B \in \mathcal{F}_{\{u,v\}}$ have a giant component X_B, containing at least $N_0/2^{d+2}$ vertices.
Indeed, the expected average degree in G_B is $Np = \Omega(\omega) \to \infty$ (and with probability $1-e^{-n^{1-o(1)}}$ we have $N = n^{1-o(1)}$) and at this value the giant component is almost all of B a.a.s. In particular, since $\mathcal{S}_B$ occurs, we have that

$$\mathbf{Pr}(|X_B| \le N_0/2^{d+2} \mid \mathcal{S}_B) \le e^{-\Omega(N_0)} \le e^{-\Omega(\delta^d n)} = o(1). \tag{8}$$

See [2] for the first inequality in (8). This can be inflated by $n^2 \cdot (2C\log n)$ to account for pairs u,v and the choice of $B \in \mathcal{F}_{\{u,v\}}$.

(B) There is an edge between X_B and $X_{B'}$ for all $B,B' \in \mathcal{F}_{\{u,v\}}$.
Indeed, the probability that there is no edge between $X_B, X_{B'}$, given (A), is at most

$$(1-p)^{N_0^2/2^{2d+2}} \le e^{-\Omega(\delta^{2d}n^2p)} \le e^{-n^{1-o(1)}}.$$

This can be inflated by $n^2 \cdot (C\log n)^2$ to account for all pairs u,v and all pairs B,B'.

(C) For each $B \in \mathcal{F}_{\{u,v\}}$, the graph diameter $\operatorname{diam}(X_B)$ (the maximum number of edges in any shortest path in X_B) satisfies

$$\mathbf{Pr}\left(\operatorname{diam}(X_B) > \frac{100\log N_0}{\log(N_0p)}\right) \le n^{-3}. \tag{9}$$

This can be inflated by $n^2 \cdot (2C\log n)$ to account for pairs u,v and the choice of $B \in \mathcal{F}_{\{u,v\}}$. Fernholz and Ramachandran [4] and Riordan and Wormald [11] gave tight estimates for the diameter of the giant component, but we need this cruder estimate with a lower probability of being exceeded. We prove this later in Lemma 5. It will be convenient for the proof of Lemma 5 to assume that $N_0p = O(\log N_0)$. There is no loss in generality because Theorem 1(b) holds a fortiori for larger p. This follows from a standard coupling argument, involving adding random edges to increase the edge probability.

Part (C) implies that with high probability, for any u,v at distance $\ge \gamma$ and all $B \in \mathcal{F}_{\{u,v\}}$ and vertices $x,y \in X_B$,

$$\operatorname{dist}(x,y) \le 200\delta \times \frac{\log N_0}{\log(N_0p)} \le \frac{200}{\log n} \times \frac{\log n - d\log\log n + \log\nu_d}{\log\omega + \log\nu_d} = o(1). \tag{10}$$

As the giant components X_B $(B \in \mathcal{F}_{u,v})$ contain in total at least $\frac{C}{\delta}\frac{N_0}{2^{d+2}} = \frac{C}{2^{d+2}}\nu_d n\delta^{d-1}$ vertices, the probability that u has no neighbor in these giant components is at most

$$(1-p)^{C\nu_d n\delta^{d-1}/2^{d+2}} \le e^{-C\nu_d np\delta^{d-1}/2^{d+2}} = n^{-\omega C\nu_d/2^{d+2}}.$$

In particular, the probability is small after multiplication by n^2, and thus a.a.s., for all pairs $x, y \in X_{n,p}$, x has a neighbor in X_B for some $B \in \mathcal{F}_{u,v}$ and y has a neighbor in $X_{B'}$ for some $B' \in \mathcal{F}_{v,u}$. Now by part (B) and equation (10), we can find a path

$$u, w_0, w_1, \dots, w_s, z_t, z_{t-1}, \dots, z_1, z_0, v$$

from u to v where the w_i's are all in some X_B for $B \in \mathcal{F}_{u,v}$ and the total Euclidean length of the path $w_0, \dots, w_s$ tends to zero with n, and the z_i's are all in some $X_{B'}$ for some $B' \in \mathcal{F}_{v,u}$, and the total Euclidean length of the path $z_0, \dots, w_t$ tends to zero with n. Meanwhile, the Euclidean segments corresponding to the three edges u, w_0, w_s, z_t, and z_0, v lie within δ of disjoint segments of the line segment from u to v, and thus have total length $\leq ||u - v|| + 6\delta$, giving

$$\operatorname{dist}(u, v) \leq ||u - v|| + 6\delta + o(1) = ||u - v|| + o(1). \tag{11}$$

We must also handle vertices $u, v \in \mathcal{X}_{n,p}$ with $||u - v|| < \gamma$. Given such a pair, we let B_u, B_v denote any choice of balls of radius γ such $\operatorname{dist}(B_u, B_v) \geq \gamma$, $\operatorname{dist}(B_u, u), \operatorname{dist}(B_v, v) \leq \gamma(\sqrt{d} + 2)$. (These bounds are chosen to make such a choice trivially possible, even when u, v are close to a corner.) Observe that we have: where C_u, C_v denote the giant components of B_u, B_v,

$$\mathbf{Pr}(\forall u, v \in \mathcal{X}_{n,p}, \exists w \in C_u, z \in C_v \text{ such that } u \sim w, v \sim z) \to 1 \tag{12}$$

with n since a.a.s we have that B_u and B_v contain at least $\nu_d n \gamma^d / 2^{d+2}$ points for all $u, v \in \mathcal{X}_{n,p}$ and we have that $1 - 2n^2(1-p)^{n \cdot \nu_d \gamma^d / 2^{d+2}} \to 1$. In particular, we can a.a.s for all pairs $u, v \in \mathcal{X}_{n,p}$ find $w \sim u$ within distance $\gamma(\sqrt{d} + 4)$ of u, $z \sim v$ within Euclidean distance $\gamma(\sqrt{d} + 4)$ of v, such that

$$\gamma \leq ||w - z|| \leq (2\sqrt{d} + 8)\gamma.$$

Now, we can use the previous case (11) to see that

$$\operatorname{dist}(u, v) \leq (2\sqrt{d} + 9)\gamma + 6\delta + o(1) = o(1). \tag{13}$$

In particular, $\operatorname{dist}(u, v) - ||u - v|| = o(1)$. ◀

We complete the proof of Theorem 1 by proving

▶ **Lemma 5.** *Suppose that $Np = \omega \to \infty, \omega = O(\log N)$ and let C_1 denote the unique giant component of size $N - o(N)$ in $G_{N,p}$, that q.s.*[2] *exists. Then for L large,*

$$\mathbf{Pr}\left(\operatorname{diam}(C_1) \geq \frac{L \log N}{\log Np}\right) \leq O(N^{-L/10}).$$

Proof. See appendix. ◀

3 Traveling among all vertices

Our first aim is to prove Theorem 3; this will be accomplished in Section 3.2, below. In fact, we will prove the following general statement, which will also be useful in the proof of Theorem 2:

[2] A sequence of events $\mathcal{E}_n$ occurs *quite surely* q.s. if $\mathbf{Pr}(\neg\mathcal{E}_n) = O(n^{-K})$ for all positive constants K.

▶ **Theorem 6.** *Let $\mathcal{Y}_1^d \subset [0,1]^d$ denote a set of points chosen from any fixed distribution, such that the cardinality $Y = |\mathcal{Y}_1^d|$ satisfies $\mathbf{E}(Y) = \mu > 0$ and $\mathbf{Pr}(Y \geq k) \leq C\rho^k$ for all k, for some $C > 0, \rho < 1$. For $t > 0$ let $\mathcal{Y}_t^d$ denote a random set of points in $[0,t]^d$ obtained from the union of t^d independent copies $\mathcal{Y}_1^d + x$ ($x \in \{0, \cdots, t-1\}^d$).*

If $p > 0$ is constant, $d \geq 2$, and $\mathcal{Y}_{t,p}^d$ denotes the random graph on $\mathcal{Y}_t^d$ with independent edge probabilities p, then $\exists \beta > 0$ (depending on p and the process generating $\mathcal{Y}_1^d$) such that

(i) $T(\mathcal{Y}_{t,p}^d) \approx \beta t^d$ *a.a.s., and*

(ii) $T(\mathcal{Y}_{t,p}^d) \leq \beta t^d + o(t^d)$ *q.s.*[3]

Note that as a probabilistic statement, Part (i) above asserts that there exists a choice for $o(1)$ (a function of t, say, tending to 0) such that $(1 - o(1))\beta t^d \leq T(\mathcal{Y}_{t,p}^d) \leq (1 + o(1))\beta t^d$ holds a.a.s. Similarly for Part (ii), the statement asserts the existence of a suitable fixed choice of $o(t^d)$ (a function of t, whose ratio to t^d tends to 0).

The restriction $\mathbf{Pr}\left(|\mathcal{Y}_1^d| \geq k\right) \leq C\rho^k$ simply ensures that we have exponential tail bounds on the number of points in a large number of independent copies of $\mathcal{Y}_1^d$:

▶ **Observation 7.** *For the total number T_n of points in n independent copies of $\mathcal{Y}_1^d$, we have for some absolute constant $A_{C,\rho} > 0$,*

$$\mathbf{Pr}(|T_n - \mu n| > \delta \mu n) < e^{-A_{C,\rho}\delta^2\mu^2 n}. \tag{14}$$

Note that the conditions on the distribution of $\mathcal{Y}_t^d$ are satisfied for a Poisson cloud of intensity 1, and it is via this case that we will derive Theorem 3. Other examples for which these conditions hold include the case where $\mathcal{Y}_t^d$ is simply a suitable grid of points, or is a random subset of a suitable grid of points in $[0,t]^d$, and we will make use of this latter case of Theorem 6 in our proof of Theorem 2.

Outline of proof of Theorem 6

Our proof uses subadditivity, but some of the standard properties of the classical case (e.g., monotonicity) fail in our setting, requiring us to use induction on d to achieve the result. For technical reasons (see also Question 4 of Section 5) Theorems 6 and 3 are given just for $d \geq 2$, and before beginning with the induction, we must carry out a separate argument to bound the length of the tour in 1 dimension.

When $d = 1$ all we can prove is an $O(n)$ bound on the length of the minimum tour. We do this by examining a natural greedy algorithm for finding a tour. This is the content of Lemma 8. After this we prove a sort of Lipschitz condition for the tour length, see Lemma 10. This will substitute for monotonicity. After this we can push ahead using subadditivity.

3.1 Bounding the expected tour length in 1 dimension

▶ **Lemma 8.** *Consider the random graph $G = G_{n,p}$ on the vertex set $[n]$ with constant p, where each edge $\{i,j\} \in E(G)$ is given length $|i - j| \in \mathbb{N}$. Let Z denote the minimum length of a Hamilton cycle in G starting at vertex 1, assuming one exists. If no such cycle exists let $Z = n^2$. Then there exists a constant A_p such that*

$$\mathbf{E}(Z) \leq A_p n \text{ and } Z \leq A_p n, \text{ q.s.}$$

[3] In this context $O(n^{-\omega(1)})$ is replaced by $O(t^{-\omega(1)})$.

We omit the proof due to space limitations.

Let us observe now that we get an upper bound $\mathbf{E}(T(\mathcal{Y}^1_{t,p})) \leq A_p t$ on the length of a tour in 1 dimension. We have

$$\mathbf{E}(T(\mathcal{Y}^1_{t,p})) = \sum_{n=0}^{\infty} \mathbf{E}\left(T(\mathcal{Y}^1_{t,p}) \big| |\mathcal{Y}^1_{t,p}| = n\right) \mathbf{Pr}(|\mathcal{Y}^1_{t,p}| = n).$$

When conditioning on $|\mathcal{Y}^1_{t,p}| = n$, we let $P_1 < P_2 < \cdots < P_n \subset [0,t]$ be the points in $\mathcal{Y}^1_{t,p}$. We choose $k \in \{0, n-1\}$ uniformly randomly and let $\xi_i = ||P_{k+i+1} - P_{k+i}||$, where the indices of the P_j are evaluated modulo n. We now have $\mathbf{E}(\xi_i) \leq \frac{2t}{n}$ for all i, and

$$\mathbf{E}\left(T(\mathcal{Y}^1_{t,p}) \big| |\mathcal{Y}^1_{t,p}| = n\right) \leq A_p n \cdot \frac{2t}{n},$$

and thus

$$\mathbf{E}\left(T(\mathcal{Y}^1_{t,p})\right) \leq 2A_p t. \tag{15}$$

3.2 The asymptotic tour length

Our proof of Theorem 6 uses recursion, by dividing the $[t]^d$ cube into smaller parts. However, since our divisions of the cube must not cross boundaries of the elemental regions $\mathcal{Y}^d_1$, we cannot restrict ourselves to subdivisions into perfect cubes (in general, the integer t may not have the divisors we like).

To this end, if $L = T_1 \times T_2 \times \cdots \times T_d$ where each T_i is either $[0,t]$ or $[0,t-1]$, we say L is a d-dimensional *near-cube* with sidelengths in $\{t-1, t\}$. For $0 \leq d' \leq d$, we define the canonical example $L^{d'}_d := [0,t]^{d'} \times [0,t-1]^{d-d'}$ for notational convenience, and let

$$\Phi^{d,d'}_p(t) = \mathbf{E}\left(T(\mathcal{Y}^d_{t,p} \cap L^{d'}_d)\right).$$

so that

$$\Phi^d_p(t) := \Phi^{d,d}_p(t) = \Phi^{d,0}_p(t+1).$$

In the unlikely event that $\mathcal{Y}^d_{t,p} \cap L^{d'}_d$ is not Hamiltonian, we take $T(\mathcal{Y}^d_{t,p} \cap L^{d'}_d) = t^{d+1}\sqrt{d}$, for technical reasons.

Our first goal is an asymptotic formula for Φ:

▶ **Lemma 9.** *There exists $\beta > 0$ such that*

$$\Phi^{d,d'}_p(t) \sim \beta t^d.$$

The proof of this is deferred until after the proof of Corollary 12 below.

The proof is by induction on $d \geq 2$. We prove the base case $d = 2$ along with the general case. We begin with a technical lemma.

▶ **Lemma 10.** *For every fixed p, d, there is a constant $F_{p,d} > 0$ such that*

$$\Phi^{d,d'}_p(t) \leq \Phi^{d,d'-1}_p(t) + F_{p,d} t^{d-1} \tag{16}$$

for all t sufficiently large. In particular, this implies that there is a constant $A_{p,d} > 0$ such that

$$\Phi^d_p(t+h) \leq \Phi^d_p(t) + A_{p,d} h t^{d-1} \tag{17}$$

for sufficiently large t and $1 \leq h \leq t$.

Proof. See appendix. ◀

Our argument is an adaptation of that in Beardwood, Halton and Hammersley [1] or Steele [12], with modifications to address difficulties introduced by the random set of available edges. First we introduce the concept of a decomposition into near-cubes. (Allowing near-cube decompositions is necessary for the end of the proof, beginning with Lemma 13). Simplifications relying on *Boundary Functionals* as in Yukich [13] do not appear to be available due to missing edges.

We say that a partition of $L_d^{d'}$ into m^d near-cubes S_α with sidelengths in $\{u, u+1\}$ indexed by $\alpha \in [m]^d$ is a *decomposition* if for each $1 \le b \le d$, there is an integer M_b such that, letting

$$f_b(a) = \begin{cases} au \text{ if } a < M_b \\ (a - M_b)(u+1) + M_b u \text{ if } a \ge M_b. \end{cases} .$$

we have that

$$S_\alpha = [f_1(\alpha_1 - 1), f_1(\alpha_1)] \times [f_2(\alpha_2 - 1), f_2(\alpha_2)] \times \cdots \times [f_d(\alpha_d - 1), f_d(\alpha_d)].$$

Observe that so long as $u \ll t$, $L_d^{d'}$ always has a decomposition into near-cubes with sidelengths in $\{u, u+1\}$. Indeed, if $t = ru - s$ for $0 \le s < u$ then we can take $M_b = s$ for $b \le d'$ and $M_b = s - 1$ for $b > d'$, unless $s = 0$, in which case $M_b = u - 1$.

First we note that tours in not-too-small near-cubes of a decomposition can be pasted together into a large tour at a reasonable cost:

▶ **Lemma 11.** *Fix $\delta > 0$, and suppose $t = mu$ for $u = t^\gamma$ for $\delta < \gamma \le 1$ ($m, u \in \mathbb{Z}$), and suppose S_α ($\alpha \in [m]^d$) is a decomposition of $L_d^{d'}$. We let $\mathcal{Y}_{t,p}^{d,\alpha} := \mathcal{Y}_{t,p}^d \cap S_\alpha$. We have*

$$T(\mathcal{Y}_{t,p}^d \cap L_d^{d'}) \le \sum_{\alpha \in [m]^d} T(\mathcal{Y}_{t,p}^{d,\alpha}) + 4m^d u\sqrt{d} \qquad \textit{with probability at least} \quad 1 - e^{-\Omega(u^d p^2)}.$$

Proof. See appendix. ◀

Linearity of expectation (and the upper bound $t^{d+1}\sqrt{d}$ on $T(\mathcal{Y}_{t,p}^d)$ when there is no tour) now gives a short-range recursive bound on $\Phi_p^d(t)$ when t factors reasonably well:

▶ **Corollary 12.** *For all large u and $1 \le m \le u^{10}$ ($m, u \in \mathcal{N}$),*

$$\Phi_p^d(mu) \le m^d(\Phi_p^d(u) + B_{p,d}u)$$

for some constant B_d. ◀

Proof of Lemma 9. Note that here we are using a decomposition of $[mu]^d$ into m^d subcubes with sidelength u; near-cubes are not required.

To get an asymptotic expression for $\Phi_p^d(t)$ we now let

$$\beta = \liminf_t \frac{\Phi_p^d(t)}{t^d}.$$

Choose u_0 large and such that

$$\frac{\Phi_p^d(u_0)}{u_0^d} \le \beta + \varepsilon$$

and then define the sequence $u_k, k \geq -1$ by $u_{-1} = u_0$ and $u_{k+1} = u_k^{10}$ for $k \geq 0$. Assume inductively that for some $i \geq 0$ that for $A_{p,d}$ as in Lemma 10 and $B_{p,d}$ as in Corollary 12,

$$\frac{\Phi_p^d(u_i)}{u_i^d} \leq \beta + \varepsilon + \sum_{j=-1}^{i-2} \left(\frac{A_{p,d}}{u_j} + \frac{B_{p,d}}{u_j^{d-1}} \right). \tag{18}$$

This is true for $i = 0$, and then for $i \geq 0$ and $0 \leq u \leq u_i$ and $d \leq m \in [u_{i-1}, u_{i+1}]$ we have

$$\begin{aligned}
\frac{\Phi_p^d(mu_i + u)}{(mu_i + u)^d} &\leq \frac{\Phi_p^d(mu_i) + A_{p,d}u(mu_i)^{d-1}}{(mu_i)^d}, && \text{from Lemma 10,} \\
&\leq \frac{m^d(\Phi_p^d(u_i) + B_{p,d}u_i) + A_{p,d}u(mu_i)^{d-1}}{(mu_i)^d}, && \text{from Corollary 12,} \quad (19) \\
&\leq \beta + \varepsilon + \sum_{j=-1}^{i-2} \left(\frac{A_{p,d}}{u_j} + \frac{B_{p,d}}{u_j^{d-1}} \right) + \frac{B_{p,d}}{u_i^{d-1}} + \frac{A_{p,d}}{m}, && \text{by induction,} \\
&\leq \beta + \varepsilon + \sum_{j=-1}^{i-1} \left(\frac{A_{p,d}}{u_j} + \frac{B_{p,d}}{u_j^{d-1}} \right). && (20)
\end{aligned}$$

Putting $m = u_{i+1}/u_i$ and $u = 0$ into (20) completes the induction. We deduce from (18) and (20) that for $i \geq 0$ we have

$$\frac{\Phi_p^d(t)}{t^d} \leq \beta + \varepsilon + \sum_{j=-1}^{\infty} \left(\frac{A_{p,d}}{u_j} + \frac{B_{p,d}}{u_j^{d-1}} \right) \leq \beta + 2\varepsilon \qquad \text{for } t \in J_i = [u_{i-1}u_i, u_i(u_{i+1}+1)] \tag{21}$$

Now $\bigcup_{i=0}^{\infty} J_i = [u_0^2, \infty]$ and since ε is arbitrary, we deduce that

$$\beta = \lim_{t \to \infty} \frac{\Phi_p^d(t)}{t^d}, \tag{22}$$

We can conclude that

$$\Phi_p^d(t) \sim \beta t^d,$$

which, together with Lemma 10, completes the proof of Lemma 9, once we show that $\beta > 0$ in (22). To this end, we let ρ denote $\mathbf{Pr}(|\mathcal{Y}_1^d| \geq 1)$, so that $\mathbf{E}(|\mathcal{Y}_t^d|) \geq \rho t^d$. We say $x \in \{0, \ldots, t-1\}^d$ is *occupied* if there is a point in the copy $\mathcal{Y}_1^d + x$. Observing that a unit cube $[0,1]^d + x$ ($x \in \{0, \ldots, t-1\}^d$) is at distance at least 1 from all but $3^d - 1$ other cubes $[0,1]^d + y$, we certainly have that the minimum tour length through $\mathcal{Y}_t^d$ is at least $\frac{\mathcal{O}}{3^d - 1}$, where where $\mathcal{O}$ is the number of occupied x. Linearity of expectation now gives that $\beta > \rho/(3^d - 1)$, completing the proof of Lemma 9. ◀

Before continuing, we prove the following much cruder version of Part (ii) of Theorem 6:

▶ **Lemma 13.** *For any fixed $\varepsilon > 0$, $T(\mathcal{Y}_{t,p}^d) \leq t^{d+\varepsilon}$ q.s.*

Proof. We let $m = \lfloor t^{1-\varepsilon/2} \rfloor$, $u = \lfloor t/m \rfloor$, and let $\{\mathcal{Y}_{\tau,p}^{d,\alpha}\}$ be a decomposition of $\mathcal{Y}_{t,p}^d$ into m^d near-cubes with sidelengths in $\{u, u+1\}$. We have that q.s. each $\mathcal{Y}_{\tau,p}^{d,\alpha}$ has (i) $\approx u^d$ points, and (ii) a Hamilton cycle H_α. We can therefore q.s. bound all $T(\mathcal{Y}_{\tau,p}^{d,\alpha})$ by $du \cdot u^d$, and Lemma 11 gives that q.s. $T(\mathcal{Y}_{t,p}^d) \leq 4dut^d + 4m^d u\sqrt{d}$. ◀

Proof of Theorem 6. We consider a decomposition $\{S_\alpha\}$ $(\alpha \in [m]^d)$ of $\mathcal{Y}_t^d$ into m^d near-cubes of side-lengths in $\{u, u+1\}$, for $\gamma = 1 - \frac{\varepsilon}{2}$, $m = \lfloor t^\gamma \rfloor$, and $u = \lfloor t/m \rfloor$.

Lemma 9 gives that

$$\mathbf{E}\, T(\mathcal{Y}_{t,p}^{d,\alpha}) \sim \beta u^d \sim \beta t^{(1-\gamma)d}.$$

Let

$$\mathcal{S}_\gamma(\mathcal{Y}_{t,p}^d) = \sum_{\alpha \in [m]^d} \min\left\{T(\mathcal{Y}_{t,p}^{d,\alpha}), 2dt^{(1-\gamma)(d+\varepsilon)}\right\}.$$

Note that $\mathcal{S}_\gamma(\mathcal{Y}_{t,p}^d)$ is the sum of $t^{\gamma d}$ identically distributed bounded random variables.

Now, since q.s. $T(\mathcal{Y}_{t,p}^{d,\alpha}) \leq 2dt^{(1-\gamma)(d+\varepsilon)}$ for all α by Lemma 13, we have that q.s. $\mathcal{S}_\gamma(\mathcal{Y}_{t,p}^d) = \sum_\alpha T(\mathcal{Y}_{t,p}^{d,\alpha})$. Applying Hoeffding's theorem we see that for any $\xi > 0$, we have

$$\mathbf{Pr}(|\mathcal{S}_\gamma(\mathcal{Y}_{t,p}^d) - m^d\, \mathbf{E}(T(\mathcal{Y}_{u,p}^d))| \geq \xi) \leq 2\exp\left(-\frac{2\xi^2}{4m^d d^2 t^{2(1-\gamma)(d+\varepsilon)}}\right).$$

Putting $\xi = t^{d\varepsilon}$ for small ε, we see that

$$\mathcal{S}_\gamma(\mathcal{Y}_{t,p}^d) = \beta t^d + o(t^d) \qquad q.s. \tag{23}$$

Note next that Lemma 11 implies that

$$T(\mathcal{Y}_{t,p}^d) \leq \mathcal{S}_\gamma(\mathcal{Y}_{t,p}^d) + \delta_2 \text{ where } \delta_2 = o(t^d) \qquad q.s. \tag{24}$$

It follows from (23) and (24) and the fact that $\mathbf{Pr}(|\mathcal{Y}_t^d| = t^d) = \Omega(t^{-d/2})$ that

$$T(\mathcal{Y}_{t,p}^d) \leq \beta t^d + o(t^d) \qquad q.s. \tag{25}$$

which proves part (ii) of Theorem 6.

Of course, we have from Lemma 9 that

$$\mathbf{E}(T(\mathcal{Y}_{t,p}^d)) = \beta t^d + \delta_1 \text{ where } \delta_1 = o(t^d), \tag{26}$$

and we show next that that this together with (24) implies part (i) of Theorem 6, that:

$$T = T(\mathcal{Y}_{t,p}^d) = \beta t^d + o(t^d) \qquad a.a.s. \tag{27}$$

We choose $0 \leq \delta_3 = o(t^d))$ such that $0 \leq \delta_2, |\delta_1| = o(\delta_3)$. Let $I = [\beta t^d - \delta_3, \beta t^d + \delta_2]$. Then we have

$$\begin{aligned}\beta t^d + \delta_1 = \mathbf{E}(T(\mathcal{Y}_{t,p}^d) \mid T(\mathcal{Y}_{t,p}^d) \geq (\beta t^d + \delta_2)\, \mathbf{Pr}(T(\mathcal{Y}_{t,p}^d) \geq \beta t^d + \delta_2)\\ + \mathbf{E}(T(\mathcal{Y}_{t,p}^d) \mid T(\mathcal{Y}_{t,p}^d) \in I)\, \mathbf{Pr}(T(\mathcal{Y}_{t,p}^d) \in I) +\\ \mathbf{E}(T(\mathcal{Y}_{t,p}^d) \mid T(\mathcal{Y}_{t,p}^{d,\alpha}) \leq \beta t^d - \delta_3)\, \mathbf{Pr}(T(\mathcal{Y}_{t,p}^d) \leq \beta t^d - \delta_3).\end{aligned}$$

Now $\varepsilon_1 = \mathbf{E}(T(\mathcal{Y}_{t,p}^d) \mid T(\mathcal{Y}_{t,p}^d) \geq \beta t^d + \delta_2)\, \mathbf{Pr}(T(\mathcal{Y}_{t,p}^d) \geq \beta t^d + \delta_2) = O(t^{-\omega(1)})$ since $|\mathcal{Y}_{t,p}^d| \leq 2d^{1/2}t^d$ and $\mathbf{Pr}(T(\mathcal{Y}_{t,p}^d) \geq \beta t^d + \delta_2) = O(t^{-\omega(1)})$, from (25).

So, if $\lambda = \mathbf{Pr}(T(\mathcal{Y}_{t,p}^d) \in I)$ then we have

$$\beta t^d + \delta_1 \leq \varepsilon_1 + (\beta t^d + \delta_2)\lambda + (\beta t^d - \delta_3)(1 - \lambda)$$

or

$$\lambda \geq \frac{\delta_1 - \varepsilon_1 + \delta_3}{\delta_2 + \delta_3} = 1 - o(1),$$

and this proves (27) completing the proof of Theorem 6. ◀

Proof of Theorem 3. We now let $\mathcal{W}_{t,p}^d$ be the graph on the set of points in $[0,t]^d$ which is the result of a Poisson process of intensity 1. Our first task is to bound the variance $\mathcal{V}(t)$ of $T(\mathcal{W}_{t,p}^d)$. Here we follow Steele's argument [12] with only small modifications. We approximate $T(\mathcal{W}_{2t,p}^d)$ as the sum over 2^d half-size cubes of $T(\mathcal{W}_{t,p}^d)$ and use this to show that $\sum_{k=1}^{\infty} \frac{\mathcal{V}(2^k t)}{(2^k t)^{2d}} \leq \infty$. This deals with n of the form $2^k t$ for some value of t and we then have to fill in the gaps.

Let $\mathcal{E}_t$ denote the event that

$$T(\mathcal{W}_{2t,p}^d) \leq \sum_{\alpha \in [2]^d} T(\mathcal{W}_{t,p}^{d,\alpha}) + 2^{d+2} t\sqrt{d}. \tag{28}$$

Observe that Lemma 11 implies that

$$\mathbf{Pr}(\neg \mathcal{E}_t) \leq e^{-\Omega(t^d p)}. \tag{29}$$

We define the random variable $\lambda(t) = T(\mathcal{W}_{t,p}^d) + 10t\sqrt{d}$, and let λ_i denote independent copies of $\lambda(t)$. Conditioning on $\mathcal{E}_t$, we have from (28) that

$$\lambda(2t) \leq \sum_{i=1}^{2^d} \lambda_i(t) - 4t\sqrt{d} \leq \sum_{i=1}^{2^d} \lambda_i(t). \tag{30}$$

In particular, (29) implies that letting $\Upsilon(t) = \mathbf{E}(\lambda(t)) = \Omega(t^d)$ (see (26)) and $\Psi(t) = \mathbf{E}(\lambda(t)^2)$, we have for sufficiently large t that

$$\begin{aligned}
\Psi(2t) &\leq \mathbf{E}\left(\left(\sum_{\alpha \in [2]^d} T(\mathcal{W}_{t,p}^{d,\alpha}) + 2^{d+2} t\sqrt{d} + 21t\sqrt{d}\right)^2\right) \\
&= \sum_{i=1}^{2^d} \mathbf{E}((\lambda_i(t) - 10t\sqrt{d})^2) + \sum_{i \neq j} \mathbf{E}(\lambda_i(t) - 10t\sqrt{d})\, \mathbf{E}(\lambda_j(t) - 10t\sqrt{d}) + \\
&\qquad + (2^{d+2} + 21)t\sqrt{d} \sum_{i=1}^{2^d} \mathbf{E}(\lambda_i(t) - 10t\sqrt{d}) + ((2^{d+2} + 21)t\sqrt{d})^2 \\
&= 2^d\, \mathbf{E}((\lambda(t) - 10t\sqrt{d})^2) + 2^d(2^d - 1)\, \mathbf{E}(\lambda(t) - 10t\sqrt{d})^2 + \\
&\qquad + 2^d(2^{d+2} + 21)t\sqrt{d}\, \mathbf{E}(\lambda(t) - 10t\sqrt{d}) + ((2^{d+2} + 21)t\sqrt{d})^2 \\
&= 2^d \Psi(t) + 2^d(2^d - 1)\Upsilon(t)^2 - \Omega(t\, \mathbf{E}(\lambda(t)) + O(t^2)) \\
&\leq 2^d \Psi(t) + 2^d(2^d - 1)\Upsilon(t)^2.
\end{aligned}$$

For

$$\mathcal{V}(t) := \mathbf{Var}(T(\mathcal{W}_{t,p}^d)) = \Psi(t) - \Upsilon(t)^2,$$

we have

$$\frac{\mathcal{V}(2t)}{(2t)^{2d}} - \frac{1}{2^d} \frac{\mathcal{V}(t)}{t^{2d}} \leq \frac{\Upsilon(t)^2}{t^{2d}} - \frac{\Upsilon(2t)^2}{(2t)^{2d}}.$$

Now with $t \geq 1$ arbitrary, summing over $2^k t$ for $k = 0, \ldots, M-1$ gives

$$\sum_{k=1}^{M} \frac{\mathcal{V}(2^k t)}{(2^k t)^{2d}} - \frac{1}{2^d} \sum_{k=0}^{M-1} \frac{\mathcal{V}(2^k t)}{(2^k t)^{2d}} \leq \frac{\Upsilon(t)^2}{t^{2d}} - \frac{\Upsilon(2^M t)^2}{(2^M t)^{2d}} \leq \frac{\Upsilon(t)^2}{t^{2d}}$$

and so, solving for the first sum, we find

$$\sum_{k=1}^{M} \frac{\mathcal{V}(2^k t)}{(2^k t)^{2d}} \leq \left(1 - \frac{1}{2^d}\right)^{-1} \left(\frac{\mathcal{V}(t)}{t^{2d}} + \frac{\Upsilon(t)^2}{t^{2d}}\right) < \infty. \tag{31}$$

Still following Steele, we let $N(t)$ be the Poisson counting process on $[0, \infty)$. We fix a random embedding $\mathcal{U}$ of $\mathcal{N}$ in $[0,1]^d$ as $u_1, u_2, \dots$ and a random graph $\mathcal{U}_p$ where each edge is included with independent probability p. We let $\mathcal{U}_{n,p}$ denote the restriction of this graph to the first n natural numbers. In particular, note that $\mathcal{U}_{N(t^d),p}$ is equivalent to $\mathcal{W}_{t,p}$, scaled from $[0,t]^d$ to $[0,1]^d$. Thus, applying Chebychev's inequality to (31) gives, in conjunction with Lemma 9, that

$$\sum_{k=0}^{\infty} \mathbf{Pr}\left(\left|\frac{t2^k T(\mathcal{U}_{N((t2^k)^d),p})}{(t2^k)^d} - \beta_{p,d}\right| > \varepsilon\right) < \infty \tag{32}$$

and so for $t > 0$ that

$$\lim_{k\to\infty} \frac{T(\mathcal{U}_{N((t2^k)^d),p})}{(t2^k)^{d-1}} = \beta_{p,d} \qquad a.s. \tag{33}$$

Now choosing some large integer ℓ, we have that (33) holds simultaneously for all the (finitely many) integers $t \in S_P = [2^\ell, 2^{\ell+1})$; and for $2^\ell \leq r \in \mathbb{R}$, we have that

$$r \in [2^k t, 2^k(t+1)) \text{ for } t \in S_\ell \text{ and some } k. \tag{34}$$

(We simply choose k such that $2^\ell \leq 2^{-k} r < 2^{\ell+1}$.)

◀

Unlike the classical case $p = 1$, in our setting, we do not have monotonicity of $T(\mathcal{U}_{n,p})$. Nevertheless, we show a kind of continuity of the tour length through $T(\mathcal{U}_{n,p})$:

▶ **Lemma 14.** *For all $\varepsilon > 0$, $\exists \delta > 0$ such that for all $0 \leq k < \delta n$, we have*

$$T(\mathcal{U}_{n+k,p}) < T(\mathcal{U}_{n,p}) + \varepsilon n^{\frac{d-1}{d}}, \qquad q.s. \tag{35}$$

Proof. See appendix. ◀

Applying Lemma 14 with $\delta = (1+\frac{1}{t})^d - 1 = O(\frac{d}{t})$ so that we have $(2^k t)^d \leq r^d \leq (2^k t)^d(1+\delta)$ by (34), and using the fact that

$$(1-2\delta)N(r^d) < N((1-\delta)r^d) < N((1+\delta)r^d) < (1+2\delta)N(r^d) \; q.s. \text{ (with respect to } r\text{)},$$

gives that for some $\varepsilon_\ell > 0$ which can be made arbitrarily small by increasing ℓ, we have q.s.

$$T(\mathcal{U}_{N(((t+1)2^k)^d),p}) - \varepsilon_\ell r^{d-1} < T(\mathcal{U}_{N(r^d),p}) < T(\mathcal{U}_{N((t2^k)^d),p}) + \varepsilon_\ell r^{d-1},$$

and so dividing by r^{d-1} and using (33) and taking limits we find that a.s.

$$\beta_{p,d} - 2\varepsilon_\ell \leq \liminf_{r\to\infty} \frac{T(\mathcal{U}_{N(r^d)})}{r^{d-1}} \leq \limsup_{r\to\infty} \frac{T(\mathcal{U}_{N(r^d)})}{r^{d-1}} \leq \beta_{p,d} + 2\varepsilon_\ell.$$

Since ℓ may be arbitrarily large, we find that

$$\lim_{r\to\infty} \frac{T(\mathcal{U}_{N(r^d)})}{r^{d-1}} = \beta_{p,d}.$$

Now the elementary renewal theorem guarantees that

$$N^{-1}(n) \sim n, \qquad a.s.$$

So we have a.s.

$$\lim_{r\to\infty} \frac{T(\mathcal{U}_{n,p})}{n^{\frac{d-1}{d}}} = \lim_{r\to\infty} \frac{T(\mathcal{U}_{N(N^{-1}(n)),p})}{(N^{-1}(n))^{\frac{d-1}{d}}} \frac{(N^{-1}(n))^{\frac{d-1}{d}}}{n^{\frac{d-1}{d}}} = \beta_{p,d} \cdot 1 = \beta_{p,d}.$$

4 The case $p(n) \to 0$

This is omitted due to space restrictions.

5 Further questions

Theorem 1 shows that there is a definite qualitative change in the diameter of $\mathcal{X}_{n,p}$ at around $p = \frac{\log^d n}{n}$, but our methods leave a $(\log\log n)^{2d}$ size gap for the thresholds.

▶ **Question 1.** *What is the precise threshold for there to be distances in $\mathcal{X}_{n,p}$ which tend to ∞? What is the precise threshold for distance in $\mathcal{X}_{n,p}$ to be arbitrarily close to Euclidean distance? What is the behavior of the intermediate regime?*

One could also analyze the geometry of the geodesics in $\mathcal{X}_{n,p}$ (Figure 1). For example:

▶ **Question 2.** *Let ℓ be the length of a random edge on the geodesic between fixed points at at constant distance in $\mathcal{X}_{n,p}$. What is the distribution of ℓ?*

Improving Theorem 2 to give an asymptotic formula for $T(\mathcal{X}_{n,p})$ is another obvious target. It may seem unreasonable to claim such a formula for all (say, decreasing) functions p; in particular, in this case, the constant in the asymptotic formula would necessarily be universal. The following, however, seems reasonable:

▶ **Conjecture 15.** *If $p = \frac{1}{n^\alpha}$ for some constant $0 < \alpha < 1$ then there exists a constant $\beta_{\alpha,d}$ such that a.a.s. $T(\mathcal{X}_{n,p}) \sim \beta_{\alpha,d}\frac{n^{\frac{d-1}{d}}}{p^{1/d}}$.*

We note that $T(\mathcal{X}_{n,1})$ is known to be remarkably well-concentrated around its mean; see, for example, the sharp deviation result of Rhee and Talagrand [10].

▶ **Question 3.** *How concentrated is the random variable $T(\mathcal{X}_{n,p})$?*

The case of where $p = o(1)$ may be particularly interesting.

Even for the case $p = 1$ covered by the BHH theorem, the constant $\beta_{1,d}$ $(d \geq 2)$ from Theorem 6 is not known. Unlike the case of $p = 1$, the 1-dimensional case is not trivial for our model. In particular, we have proved Theorems 3 and 2 only for $d \geq 2$. We have ignored the case $d = 1$ not because we consider the technical problems insurmountable, but because we hope that it may be possible to prove a stronger result for $d = 1$, at least for the case of constant p.

▶ **Question 4.** *Determine an explicit constant $\beta_{p,1}$ as a function of (constant) p such that for $d = 1$,*

$$\lim_{n\to\infty} T(\mathcal{X}_{n,p}) = \beta_{p,1}.$$

Our basic motivation has been to understand the constraint imposed on travel among random points by the restriction set of traversable edges which is chosen randomly independently of the geometry of the underlying point-set. While the Erdős-Rényi-Gilbert model is the prototypical example of a random graph, other models such as the Barabási-Albert preferential attachment graph have received wide attention in recent years, due to properties (in particular, the distribution of degrees) they share with real-world networks.

▶ **Question 5.** *If the preferential attachment graph is embedded randomly in the unit square (hypercube), what is the expected diameter? What is the expected size of a minimum-length spanning tree?*

References

1 J. Beardwood, J. H. Halton, and J. M. Hammersley. The shortest path through many points. *Mathematical Proceedings of the Cambridge Philosophical Society*, 55:299–327, 1959.

2 B. Bollobás. *Random Graphs, Second Edition.* Cambridge University Press, 2001.

3 B. Bollobás, T. Fenner, and A.M. Frieze. An algorithm for finding hamilton paths and cycles in random graphs. *Combinatorica*, 7:327–341, 1987.

4 D. Fernholz and V. Ramachandran. The diameter of sparse random graphs. *Random Structures and Algorithms*, 31:482–516, 2007.

5 Y. Gurevich and S. Shelah. Expected computation time for hamiltonian path problem. *SIAM Journal on Computing*, 16:486–502, 1987.

6 R. M. Karp. Probabilistic analysis of partitioning algorithms for the traveling-salesman problem in the plane. *Mathematics of Operations Research*, 2:209–244, 1977.

7 A. Mehrabian. A randomly embedded random graph is not a spanner. In *Proceedings of the 23rd Canadian Conference on Computational Geometry (CCCG 2011)*, pages 373–374, 2011.

8 A. Mehrabian and N. Wormald. On the stretch factor of randomly embedded random graphs. *Discrete & Computational geometry*, 49:647–658, 2013.

9 P. M. Penrose. *Random Geometric Graphs.* Oxford University Press, 2003.

10 W. Rhee and M. Talagrand. A sharp deviation inequality for the stochastic traveling salesman problem. *The Annals of Probability*, 17:1–8, 1989.

11 O. Riordan and N. Wormald. The diameter of sparse random graphs. *Combinatorics, Probability and Computing*, 19:835–926, 2010.

12 J. M. Steele. Subadditive euclidean functionals and nonlinear growth in geometric probability. *The Annals of Probability*, 9:365–376, 1981.

13 J. Yukich. *Probability Theory of Classical Euclidean Optimization Problems.* Springer, 1991.

A Proof of Lemma 5

Let $\mathcal{B}(k)$ be the event that there exists a set S of k vertices in $G_{N,p}$ that induces a connected subgraph and in which more than half of the vertices have less than $\omega/2$ neighbors outside S. Then for $k = o(N)$ we have

$$\begin{align}
\mathbf{Pr}(\mathcal{B}(k)) &\leq \binom{N}{k} p^{k-1} k^{k-2} 2^k\, \mathbf{Pr}(Bin(N-k,p) \leq \omega/2)^{k/2} \tag{36}\\
&\leq \frac{e^k \omega^k}{pk^2} 2^k \left(e^{-((N-k)p-\omega/2)^2/(2(N-k)p)}\right)^{k/2}, \text{ from (6) with } \varepsilon = 1 - \frac{\omega}{2(N-k)p}, \tag{37}\\
&\leq \frac{e^k \omega^k}{pk^2} 2^k \left(e^{-(.99\omega-\omega/2)^2/2\omega}\right)^{k/2} \tag{38}\\
&\leq p^{-1}(2e\omega e^{-\omega/20})^k \leq Ne^{-k\omega/21}. \tag{39}\\
& \tag{40}
\end{align}$$

Explanation of (36): $\binom{N}{k}$ bounds the number of choices for S. We then choose a spanning tree T for S in k^{k-2} ways. We multiply by p^{k-1}, the probability that T exists. We then choose half the vertices X of S in at most 2^k ways and then multiply by the probability that each $x \in X$ has at most $\omega/2$ neighbors in $[N] \setminus S$.

If $\kappa = \kappa(L) = \frac{L \log N}{\log Np}$ then (39) implies that $\mathbf{Pr}(\mathcal{B}(\kappa)) \leq N^{1-L}$.

Next let $\mathcal{D}(k)=\mathcal{D}_N(k)$ be the event that there exists a set S of size k for which the number of edges $e(S)$ contained in S satisfies $e(S)\geq 2k$. Then,

$$\mathbf{Pr}(\mathcal{D}(k))\leq \binom{N}{k}\binom{\binom{k}{2}}{2k}p^{2k}\leq \left(\frac{Ne}{k}\cdot\left(\frac{ke\omega}{2N}\right)^2\right)^k=\left(\frac{ke^3\omega^2}{4N}\right)^k.$$

Since $\omega=O(\log n)$ we have that q.s.

$$\nexists k\in[\kappa(1),N^{3/4}] \text{ such that } \mathcal{D}(k) \text{ occurs.} \tag{41}$$

Now let $\mathcal{B}(k_1,k_2)=\bigcup_{k=k_1}^{k_2}\mathcal{B}(k)$ and $\mathcal{D}(k_1,k_2)=\bigcup_{k=k_1}^{k_2}\mathcal{D}(k)$, and suppose that

$$\mathcal{B}(k_1,k_2)\cup\mathcal{D}(k_1,k_2) \text{ does not occur,} \tag{42}$$

where $k_1=\kappa(L/4)$ and $k_2=N^{3/4}$. Fix a pair of vertices v,w and define sets $S_0,S_1,S_2,\dots$ where S_i is the set of vertices at distance i from v. If there is no $i\leq k_1$ with $w\in S_i$ then we must have $S_{k_1}\neq\emptyset$ and $|S_{\leq k_1}|\geq k_1$ where $S_{\leq t}=\bigcup_{i=0}^t S_i$ for $t\geq 0$. This is because $v,w\in C_1$ and C_1 is connected and so $|S_{\leq i+1}|\geq|S_{\leq i}|+1$. We also see that $k_1\leq|S_{\leq t}|\leq N^{3/4}$ implies that $|S_{t+1}|\geq\omega|S_{\leq t}|/10$. Indeed, if $|S_{t+1}|<\omega|S_{\leq t}|/10$ then $S_{\leq t+1}$ has at most $(\omega+10)|S_{\leq t}|/10$ vertices and more than $\omega|S_{\leq t}|/4$ edges, contradiction.

Thus if L is large, then we find that there exists $t\leq k_1+\kappa(3/4)\leq N^{3/4}$ such that $|S_{\leq t}|\geq N^{3/4}$ and so also that $|S_t|\geq(1-o(1))N^{3/4}$. Now apply the same argument from w to create sets $T_0,T_1,\dots,T_s$, where either we reach v or find that $|T_s|\geq N^{3/4}$ where $s\leq k_1+\kappa(3/4)$. At this point the edges between S_t and T_s are unconditioned and the probability there is no $S_t:T_s$ edge is at most $(1-p)^{N^{3/2}}=O(e^{-\Omega(N^{1/2})})$.

B Proof of Lemma 10

We let S denote the subgraph of $\mathcal{Y}^d_{t,p}\cap L^{d'}_d$ induced by the difference $L^{d'}_d\setminus L^{d'-1}_d$.

By ignoring the d'th coordinate of S, we obtain the $(d-1)$ dimensional set $\pi(S)$, for which induction on d (or equation (15) if $d=2$) implies an expected tour $T(S)$ of length $\Phi_p^{d-1,d'-1}(t)\leq\beta_p^{d-1}t^{d-1}$, and so changing notation, we can write

$$\Phi_p^{d-1,d'-1}(t)\leq D_{p,d-1}t^{d-1}.$$

We have that

$$\mathbf{E}(T(S))\leq\mathbf{E}(T(\pi(S))+d^{1/2}\,\mathbf{E}(|\pi(S)|)\leq D_{p,d-1}t^{d-1}+d^{1/2}t^{d-1}.$$

The first inequality stems from the fact that the points in $L^{d'}_d\setminus L^{d'-1}_d$ have a d' coordinate in $[t-1,t]$.

Now if $\mathcal{Y}^d_{t,p}\cap L^{d'-1}_d$ and S are both Hamiltonian, then we have

$$T(\mathcal{Y}^d_{t,p}\cap L^{d'}_d)\leq T(\mathcal{Y}^d_{t,p}\cap L^{d'-1}_d)+T(S)+O_d(t) \tag{43}$$

which gives us the Lemma, by linearity of expectation. We have (43) because we can patch together the minimum cost Hamilton cycle H in $\mathcal{Y}^d_{t,p}\cap L^{d'-1}_d$ and the minimum cost path P in S as follows: Let u_1,v_1 be the endpoints of P. If there is an edge u,v of H such that $(u_1,u),(v_1,v)$ is an edge in $\mathcal{Y}^d_{t,p}$ then we can create a cycle H_1 through $\mathcal{Y}^d_{t,p}\cap L^{d'-1}_d\cup P$ at an extra cost of at most $2d^{1/2}t$. The probability there is no such edge is at most $(1-p^2)^{t/2}$, which is negligible given the maximum value of $T(\mathcal{Y}^d_{t,p}\cap L^{d'}_d)$.

On the other hand, because p is a constant, the probability that either of $\mathcal{Y}_{t,p}^d \cap L_d^{d'-1}$ or S is not Hamiltonian is exponentially small in t, (see for example [5]), which is again negligible given the maximum value of $T(\mathcal{Y}_{t,p}^d \cap L_d^{d'})$. This completes the proof of (16).

To obtain (17) we use (16) to write

$$\begin{aligned}\Phi_p^{d,d}(t+h) &\leq \Phi_p^{d,0}(t+h) + dF_{p,d}(t+h)^{d-1} = \Phi_p^d(t+h-1) + dF_{p,d}(t+h)^{d-1} \\ &\leq \Phi_p^d(t) + dF_{p,d}\sum_{i=0}^{h}(t+i)^{d-1}.\end{aligned}$$

C Proof of Lemma 11

Let $\mathcal{B}, \mathcal{C}$ denote the events

$$\begin{aligned}\mathcal{B} &= \left\{\exists \alpha : \mathcal{Y}_{t,p}^{d,\alpha} \text{ is not Hamiltonian}\right\}, \\ \mathcal{C} &= \left\{\exists \alpha : \left| |\mathcal{Y}_{t,p}^{d,\alpha}| - u^d \right| \geq \delta u^d \right\},\end{aligned}$$

and let $\mathcal{E} = \mathcal{B} \cup \mathcal{C}$.

Now $\mathbf{Pr}(\mathcal{B}) \leq m^d e^{-\Omega(u^d p)}$ and, by Observation 7, $\mathbf{Pr}(\mathcal{C}) \leq m^d e^{-\Omega(u^d)}$ and so $\mathbf{Pr}(\mathcal{E}) \leq e^{-\Omega(u^d p)}$. Assume therefore that $\neg\mathcal{E}$ occurs. Each subcube S_α will contain a minimum length tour H_α. We now order the subcubes $\{S_\alpha\}$ as $T_1, \ldots, T_{m^d}$, such that for $S_\alpha = T_i$ and $S_{\alpha'} = T_{i+1}$, we always have that the Hamming distance between α and α' is 1. Our goal is to inductively assemble a tour through the subcubes $T_1, T_2, \ldots, T_j$ from the smaller tours H_α with a small number of additions and deletions of edges.

Assume inductively that for some $1 \leq j < m^d$ we have added and deleted edges and found a single cycle C_j through the points in $T_1, \ldots, T_j$ in such a way that (i) the added edges have total length at most $4\sqrt{d}ju$ and (ii) we delete one edge from $\tau(T_1)$, $\tau(T_j)$ and two edges from each $\tau(T_i), 2 \leq i \leq j-1$. To add the points of T_{j+1} to create C_{j+1} we delete one edge (u,v) of $\tau(T_j) \cap C_j$ and one edge (x,y) of $\tau(T_{j+1})$ such that both edges $\{u,x\}, \{v,y\}$ are in the edge set of $\mathcal{Y}_{t,p}^d$. Such a pair of edges will satisfy (i) and (ii) and the probability we cannot find such a pair is at most $(1-p^2)^{(u^d/2-1)u^d/2}$. Thus with probability at least $1 - e^{\Omega(u^d p^2)}$ we build the cycle C_{m^d} with a total length of added edges $\leq 4\sqrt{d}m^d u$.

D Proof of Lemma 14

We consider cases according to the size of k.

Case 1: $k \leq n^{\frac{1}{3}}$. Note that we have $T(\mathcal{U}_{n+1,p}) < T(\mathcal{U}_{n,p}) + \sqrt{d}$ q.s., since we can q.s. find an edge in the minimum tour though $\mathcal{U}_{n,p}$ whose endpoints are both adjacent to $(n+1)$. $n^{\frac{1}{3}}$ applications of this inequality now give (35).

Case 2: $k > n^{\frac{1}{3}}$. In this case the restriction $\mathcal{R}$ of $\mathcal{U}_{n+k,p}$ to $\{n+1, \ldots, k\}$ is q.s. (with respect to n) Hamiltonian [3]. In particular, by Theorem 6, we can q.s. find a tour T though $\mathcal{R}$ of length $\leq 2\beta_p^d k^{\frac{d-1}{d}}$. Finally, there are q.s., edges $\{x,y\}$ and $\{w,z\}$ on the minimum tours through $\mathcal{U}_{n,p}$ and $\mathcal{R}$, respectively, such that $x \sim w$ and $y \sim z$ in $\mathcal{U}_{n+k,p}$, giving a tour of length

$$T(\mathcal{U}_{n+k,p}) \leq T(\mathcal{U}_{n,p}) + 2\beta_{p,d}k^{\frac{d-1}{d}} + 4\sqrt{d}.$$

The Minrank of Random Graphs*

Alexander Golovnev[1], Oded Regev[2], and Omri Weinstein[3]

1 **Courant Institute of Mathematical Sciences, New York University, New York, NY, USA**
golovnev@cims.nyu.edu
2 **Courant Institute of Mathematical Sciences, New York University, New York, NY, USA**
regev@cims.nyu.edu
3 **Columbia University, New York, NY, USA**
omri@cs.columbia.edu

Abstract

The *minrank* of a directed graph G is the minimum rank of a matrix M that can be obtained from the adjacency matrix of G by switching some ones to zeros (i.e., deleting edges) and then setting all diagonal entries to one. This quantity is closely related to the fundamental information-theoretic problems of (linear) *index coding* (Bar-Yossef et al., FOCS'06), network coding and distributed storage, and to Valiant's approach for proving superlinear circuit lower bounds (Valiant, Boolean Function Complexity '92).

We prove tight bounds on the minrank of directed Erdős-Rényi random graphs $G(n,p)$ for all regimes of $p \in [0,1]$. In particular, for any constant p, we show that $\mathsf{minrk}(G) = \Theta(n/\log n)$ with high probability, where G is chosen from $G(n,p)$. This bound gives a near quadratic improvement over the previous best lower bound of $\Omega(\sqrt{n})$ (Haviv and Langberg, ISIT'12), and partially settles an open problem raised by Lubetzky and Stav (FOCS '07). Our lower bound matches the well-known upper bound obtained by the "clique covering" solution, and settles the linear index coding problem for random graphs.

Finally, our result suggests a new avenue of attack, via derandomization, on Valiant's approach for proving superlinear lower bounds for logarithmic-depth semilinear circuits.

1998 ACM Subject Classification E.4 Coding and Information Theory

Keywords and phrases circuit complexity, index coding, information theory

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.46

1 Introduction

In information theory, the *index coding* problem [5, 4] is the following: A sender wishes to *broadcast* over a noiseless channel an n-symbol string $x \in \mathbb{F}^n$ to a group of n receivers $R_1, \ldots, R_n$, each equipped with some *side information*, namely, a subvector x_{K_i} of x indexed by a subset $K_i \subseteq \{1, \ldots, n\}$. The index coding problem asks what is the minimum length m of a broadcast message that allows each receiver R_i to retrieve the ith symbol x_i, given his side-information x_{K_i} and the broadcasted message. The side information of the receivers can be modeled by a directed graph $\mathcal{K}_n$, in which R_i observes the symbols $K_i := \{x_j : (i,j) \in$

* This work was partially supported by the Simons Collaboration on Algorithms and Geometry and by the National Science Foundation (NSF) under Grant No. CCF-1320188. The work was done when Omri Weinstein was supported by a Simons Junior fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 46; pp. 46:1–46:13
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

$E(\mathcal{K}_n)\}$. $\mathcal{K}_n$ is sometimes called the *knowledge graph.* A canonical example is where $\mathcal{K}_n$ is the complete graph (with no self-loops) on the vertex set $[n]$, i.e., each receiver observes all but his own symbol. In this simple case, broadcasting the sum $\sum_{i=1}^{n} x_i$ (in $\mathbb{F}$) allows each receiver to retrieve his own symbol, hence $m = 1$.

This problem is motivated by applications to distributed storage [3], on-demand video streaming (ISCOD, [6]) and wireless networks (see, e.g., [27]), where a typical scenario is that clients miss information during transmissions of the network, and the network is interested in minimizing the retransmission length by exploiting the side information clients already possess. In theoretical computer science, index coding is related to some important communication models and problems in which players have overlapping information, such as the *one-way* communication complexity of the index function [17] and the more general problem of *network coding* [1, 10]. Index coding can also be viewed as an interesting special case of nondeterministic computation in the (notoriously difficult to understand) multiparty *Number-On-Forehead* model, which in turn is a promising approach for proving data structure and circuit lower bounds [20, 21, 16]. The minimum length of an index code for a given graph has well-known relations to other important graph parameters. For instance, it is bounded from below by the size of the maximum independent set, and it is bounded from above by the clique-cover number ($\chi(\bar{G})$) since for every clique in G, it suffices to broadcast a single symbol (recall the example above). The aforementioned connections also led to algorithmic connections (via convex relaxations) between the computational complexity of graph coloring and that of computing the minimum index code length of a graph [9].

In the context of circuit lower bounds, Riis [22] observed that a certain index coding problem is equivalent to the so-called *shift conjecture* of Valiant [25] (see Subsection 1.1 below). If true, this conjecture would resolve a major open problem of proving superlinear size lower bound for logarithmic-depth circuits.

When the encoding function of the index code is *linear* in x (as in the example above), the corresponding scheme is called a *linear index code.* In their seminal paper, Bar-Yossef et al. [4] showed that the minimum length m of a *linear* index code is characterized precisely by a parameter of the knowledge graph $\mathcal{K}_n$ called the *minrank* ($\mathsf{minrk}_{\mathbb{F}}(\mathcal{K}_n)$), first introduced by Haemers [12] in the context of Shannon capacity of graphs.[1] Informally, $\mathsf{minrk}_{\mathbb{F}}(\mathcal{K}_n)$ is the minimum rank (over $\mathbb{F}$) of an $n \times n$ matrix M that "represents" $\mathcal{K}_n$ in the sense that M contains a zero in all entries corresponding to *non-edges*, and non-zero entries on the diagonal. Entries corresponding to edges are arbitrary. (Over $\mathbb{F}_2$ this is equivalent to being the adjacency matrix of a subgraph of $\mathcal{K}_n$, with diagonal entries set to one.) Note that without the "diagonal constraint", the above minimum would trivially be 0, and indeed this constraint is what makes the problem interesting and hard to analyze. While linear index codes are in fact optimal for a large class of knowledge graphs (including directed acyclic graphs, perfect graphs, odd "holes" and odd "anti-holes" [4]), there are examples where non-linear codes outperform their linear counterparts [18]. In the same paper, Lubetzky and Stav [18] posed the following question about *typical* knowledge graphs, namely,

What is the minimum length index code for a random knowledge graph $\mathcal{K}_n = \mathcal{G}_{n,p}$?

Here, $\mathcal{G}_{n,p}$ denotes a random Erdős-Rényi directed graph, i.e., a graph on n vertices in which each arc is taken independently with probability p. In this paper, we partially answer this open problem by determining the optimal length of *linear* index codes for such graphs. In

[1] To be precise, this holds only for graphs without self-loops. We will ignore this minor issue in this paper as it will not affect any of our results.

other words, we prove a tight lower bound on the minrank of $\mathcal{G}_{n,p}$ for all values of $p \in [0, 1]$. In particular,

▶ **Theorem 1** (Main theorem, informal). *For any constant $0 < p < 1$ and any field $\mathbb{F}$ of cardinality $|\mathbb{F}| < n^{O(1)}$, it holds with high probability that*

$$\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p}) = \Theta\left(\frac{n}{\log n}\right) .$$

The formal quantitative statement of our result can be found in Corollary 11 below. We note that our general result (see Theorem 10) extends beyond the constant regime to *subconstant* values of p, and this feature of our lower bound is crucial for potential applications of our result to circuit lower bounds (we elaborate on this in the next subsection). Theorem 1 gives a near quadratic improvement over the previously best lower bound of $\Omega(\sqrt{n})$ [18, 14], and settles the linear index coding problem for random knowledge graphs, as an $O_p(n/\log n)$ linear index coding scheme is achievable via the clique-covering solution (see Section 1.2).

In the following subsection, we propose a concrete (yet admittedly still quite challenging) approach for proving superlinear circuit lower bounds based on a potential "derandomization" of Theorem 1.

1.1 Connections to circuit lower bounds for semilinear circuits

1.1.1 General log-depth circuits

In his seminal line of work, Valiant [23, 24, 25] proposed a path for proving superlinear lower bounds on the size of circuits with logarithmic depth, one of the main open questions in circuit complexity. Informally speaking, Valiant's "depth reduction" method [23, 26] allows one to reduce any circuit of size $O(n)$ and depth $O(\log n)$ (with n inputs and n outputs), to a new circuit with the same inputs and outputs, where now each output gate is an (arbitrary) Boolean function of (*i*) at most n^{ε} inputs (for any constant ϵ) which are "hard-wired" to this output gate, and (*ii*) an additional fixed set of $m = O_{\varepsilon}(n/\log\log n)$ "common bits" $b_1(x), \ldots, b_m(x)$ which in general may be arbitrary Boolean functions of the input $x = x_1, \ldots, x_n$. Therefore, if one could exhibit a function that cannot be computed in this model using $O(n/\log\log n)$ common bits, this would imply a superlinear circuit lower bound for logarithmic depth circuits.

Valiant [25] proposed a concrete candidate hard function for this new model, namely the function whose input is an n-bit string x and a number $i \in \{0, \ldots, n-1\}$ and whose output is the ith cyclic shift of x. Valiant conjectured that no "pre-wired" circuit as above can realize *all* n cyclic shifts using $m = O(n/\log\log n)$ common bits (in fact, Valiant postulated that $m = \Omega(n)$ common bits are required, and this still seems plausible). This conjecture is sometimes referred to as *Valiant's shift conjecture*. As noted earlier in the introduction, Riis [22] observed that a certain index coding problem is equivalent to this conjecture. Let $G = (V, A)$ be a directed graph, and $i \in \{0, \ldots, n-1\}$. We denote by G^i the graph with vertex set V and arc set $A^i = \{(u, v + i (\text{mod } n)) : (u, v) \in A\}$. Riis [22] showed that the following conjecture is equivalent to Valiant's shift conjecture:

▶ **Conjecture 2.** *There exists $\varepsilon > 0$ such that for all sufficiently large n and every graph G on n vertices with max-out-degree at most n^{ε}, there exists a shift i such that the minimum length of an index coding scheme for G^i (over $\mathbb{F}_2$) is $\omega(n/\log\log n)$.*

1.1.2 Semilinear log-depth circuits

Let us consider a function $f(x, p)$ whose input is partitioned into two parts, $x \in \{0,1\}^k$ and $p \in \{0,1\}^t$. We say that the function f is *semilinear* if for every fixed value of $p = p_0$, the function $f(x, p_0)$ is a linear function (over $\mathbb{F}_2$) of x. The class of semilinear functions is quite rich, and includes for instance bilinear functions in x and p (such as matrix multiplication) and permutations $\pi_p(x)$ of x that may depend arbitrarily on p. A circuit G is called *semilinear* if for every fixed value of $p = p_0$, one can assign linear functions to the gates of G, so that G computes $f(x, p_0)$. So it is only the circuit's topology that is fixed, and the linear functions computed by the gates may depend arbitrarily on p.

It is easy to see that a semilinear function with a one-bit output can always be computed by a linear-size log-depth semilinear circuit (namely, the full binary tree). However, if we consider semilinear functions with $O(n)$ output bits, then the semilinear circuit complexity of a random function is $\Omega(n^2/\log n)$ with high probability. It is an open problem to prove a superlinear lower bound against log-depth semilinear circuits [21]. This would follow from the semilinear variant of Valiant's shift conjecture, which is equivalent to the following slight modification of Conjecture 2 [21, 22].

▶ **Conjecture 3.** *There exists $\varepsilon > 0$ such that for all sufficiently large n and every graph G on n vertices with max-out-degree at most n^ε, there exists a shift i such that the minimum length of a* linear *index coding scheme for G^i (over $\mathbb{F}_2$) is $\omega(n/\log\log n)$. Equivalently,*

$$\forall\, G \text{ of out-degrees at most } n^\varepsilon \;\; \exists\;\, i \in [n] \;\; \mathsf{minrk}_2(G^i) = \omega(n/\log\log n)\ .$$

Theorem 1 (and the more precise concentration bound we prove in Theorem 10) asserts that with high probability, a graph chosen from $\mathcal{G}_{n,p}$ (with $p = n^{\varepsilon-1}$ for the expected degree of each vertex to be n^ε) has minrank $\Omega(n)$. Conjecture 3 would follow from a "derandomization" of Theorem 1 in which we replace the distribution $\mathcal{G}_{n,p}$ with a random shift of an arbitrary given graph of the right degree. In fact, for the purpose of circuit lower bounds, one could replace cyclic shifts with any (efficiently computable) set of at most $\exp(O(n))$ permutations. (Since the permutation itself is part of the input, its description size must be linear in n.)

1.1.3 Semilinear series-parallel circuits

Finally, we mention one last circuit class for which the above "derandomization" approach might be easier. Here we replace the depth restriction by another restriction on the topology of the circuit. Namely, a circuit $G = (V, A)$ is called *Valiant series-parallel (VSP)*, if there is a labeling of its vertices $l\colon V \to \mathbb{R}$, such that for every arc $(u, v) \in A$, $l(u) < l(v)$, but there is no pair of arcs $(u, v), (u', v') \in A$, such that $l(u) < l(u') < l(v) < l(v')$. Most of the known circuit constructions (i.e., circuit upper bounds) are VSP circuits. Thus, it is also a big open question in circuit complexity to prove a superlinear lower bound on the size of semilinear VSP circuits (of arbitrary depth).

Valiant [23], Calabro [8], and Riis [22] show that in order to prove a superlinear lower bound for semilinear VSP circuits, it suffices to show that for a sufficiently large *constant d*, for every graph G of max-out-degree at most d, the minrank of one of its shifts is at least $n/100$. We note that Theorem 1 for this regime of $p = d/n$ gives a lower bound of $n/20$. Thus, derandomization of the theorem in this regime would imply a superlinear lower bound. Note that in the case of $p = O(n^{-1})$, the entropy of a random graph is only $O(n \log n)$ bits, hence, information-theoretically it seems easier to derandomize than the case of $p = n^{\varepsilon-1}$.

1.2 Proof overview of Theorem 1

In [18], Lubetzky and Stav showed that for any field $\mathbb{F}$ and a directed graph G,

$$\mathsf{minrk}_{\mathbb{F}}(G) \cdot \mathsf{minrk}_{\mathbb{F}}(\bar{G}) \geq n \ .$$

This inequality gives a lower bound of $\Omega(\sqrt{n})$ on the expected value of the minrank of $\mathcal{G}_{n,1/2}$. (Indeed, the random variables $\mathcal{G}_{n,1/2}$ and $\bar{\mathcal{G}}_{n,1/2}$ have identical distributions). Since $\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p})$ is monotonically non-increasing in p, the same bound holds for any $p \leq 1/2$. Haviv and Langberg [14] improved this result by proving a lower bound of $\Omega(\sqrt{n})$ for all constant p (and not just $p \leq 1/2$), and also by showing that the bound holds with high probability.

We now outline the main ideas of our proof. For simplicity we assume that $\mathbb{F} = \mathbb{F}_2$ and $p = 1/2$. To prove that $\mathsf{minrk}_2(\mathcal{G}_{n,p}) \geq k$, we need to show that with high probability, $\mathcal{G}_{n,p}$ has no representing matrix (in the sense of Definition 4) whose rank is less than k.

As a first attempt, we can show that any *fixed* matrix M with 1s on the diagonal of rank less than k has very low probability of representing a random graph in $\mathcal{G}_{n,p}$, and then apply a union bound over all such matrices M. Notice that this probability is simply 2^{-s+n}, where s is the sparsity of M (i.e., the number of non-zero entries) and the n is to account for the diagonal entries. Moreover, we observe that the sparsity s of any rank-k matrix with 1s on its main diagonal must be[2] at least $\approx n^2/k$. Finally, since the number of $n \times n$ matrices of rank k is $\approx 2^{2nk}$ (as a rank-k matrix can be written as a product of $n \times k$ by $k \times n$ matrices, which requires $2nk$ bits to specify), by a union bound, the probability that $\mathcal{G}_{n,p}$ contains a subgraph of rank $< k$ is bounded from above by (roughly) $2^{2nk} \cdot (1/2)^{n^2/k}$, which is $\ll 1$ for $k = O(\sqrt{n})$. This recovers the previous $\Omega(\sqrt{n})$ lower bound of [14] (for all constant p, albeit with a much weaker concentration bound).

To see why this argument is "stuck" at $\sqrt{n}$, we observe that we are not overcounting and indeed, there are $2^{n^{3/2}}$ matrices of rank $k \approx n^{1/2}$ and sparsity $s \approx n^{3/2}$. For instance, we can take the rank $n^{1/2}$ matrix that consists of $n^{1/2}$ diagonal $n^{1/2} \times n^{1/2}$ blocks of 1s (a disjoint union of $n^{1/2}$ equal-sized cliques), and replace the first $n^{1/2}$ columns with arbitrary values. Each such matrix has probability $2^{-n^{3/2}}$ of representing $\mathcal{G}_{n,p}$ (because of its sparsity) and there are $2^{n^{3/2}}$ of them, so the union bound breaks for $k = \Omega(\sqrt{n})$.

In order to go beyond $\sqrt{n}$, we need two main ideas. To illustrate the first idea, notice that in the above example, even though individually each matrix has probability $2^{-n^{3/2}}$ of representing $\mathcal{G}_{n,p}$, these "bad events" are highly correlated. In particular, each of these events implies that $\mathcal{G}_{n,p}$ must contain $n^{1/2} - 1$ disjoint cliques, an event that happens with roughly the same probability $2^{-n^{3/2}}$. Therefore, we see that the probability that the *union* of these bad events happens is only $2^{-n^{3/2}}$, greatly improving on the naive union bound argument. (We remark that this idea of "bunching together related events" is reminiscent of the chaining technique as used, e.g., in analyzing Gaussian processes.) More generally, the first idea (and also centerpiece) of our proof is Lemma 9, which shows that every matrix must contain a "nice" submatrix (in a sense to be defined below). The second and final idea, described in the next paragraph, will be to bound the number of "nice" submatrices, from which the proof would follow by a union bound over all such submatrices.

[2] To see why, notice that any maximal linearly independent set of columns must "cover" all coordinates, i.e., there must not be any coordinate that is zero in all vectors, as otherwise we could take the column vector corresponding to that coordinate and it would be linearly independent of our set (due to the nonzero diagonal) in contradiction to maximality. Assuming all columns have roughly the same number of 1s, we obtain that each column has at least n/k 1s, leading to the claimed bound. See Lemma 8 for the full proof.

Before defining what we mean by "nice", we mention the following elementary yet crucial fact in our proof: Every rank k matrix is uniquely determined by specifying some k linearly independent rows, and some k linearly independent columns (i.e., a row basis and a column basis) including the indices of these rows and columns (see Lemma 6). This lemma implies that we can encode a matrix using only $\approx s_{basis} \cdot \log n$ bits, where s_{basis} is the minimal sparsity of a pair of row and column bases that are guaranteed to exist. This in turn implies that there are only $\approx 2^{s_{basis} \log n}$ such matrices. Now, since the average number of 1s in a row or in a column of a matrix of sparsity s is s/n, one might hope that such a matrix contains a pair of row and column bases of sparsity $k \cdot (s/n)$, and this is precisely our definition of a "nice" matrix. (Obviously, not all matrices are nice, and as the previous example shows, there are lots of "unbalanced" matrices where the nonzero entries are all concentrated on a small number of columns, hence they have no sparse column basis even though the average sparsity of a column is very low; this is exactly why we need to go to submatrices.)

To complete this overview, notice that using the bound on the number of "nice" matrices, the union bound yields

$$2^{ks\log(n)/n} \cdot (1/2)^s,$$

so one could set the rank parameter k to be as large as $\Theta(n/\log n)$ and the above expression would still be $\ll 1$. A similar bound holds for nice submatrices, completing the proof.

2 Preliminaries

For an integer n, we denote the set $\{1, \ldots, n\}$ by $[n]$. For an integer n and $0 \leq p \leq 1$, we denote by $\mathcal{G}_{n,p}$ the probability space over the directed graphs on n vertices where each arc is taken independently with probability p.

For a directed graph G, we denote by $\chi(G)$ the chromatic number of the undirected graph that has the same set of vertices as G, and an edge in place of every arc of G. By $\bar{G}$ we mean a directed graph on the same set of vertices as G that contains an arc if and only if G does not contain it.[3]

Let $\mathbb{F}$ be a finite field. For a vector $v \in \mathbb{F}^n$, we denote by v^j the jth entry of v, and by $v^{\leq j} \in \mathbb{F}^j$ the vector v truncated to its first j coordinates. For a matrix $M \in \mathbb{F}^{n\times n}$ and indices $i, j \in [n]$, let $M_{i,j}$ be the entry in the ith row and jth column of M, $\mathrm{Col}_i(M)$ be the ith column of M, $\mathrm{Row}_i(M)$ be the ith row of M, and $\mathsf{rk}(M)$ be the rank of M over $\mathbb{F}$.

By a *principal submatrix* we mean a submatrix whose set of row indices is the same as the set of column indices. By the *leading principal submatrix* of size k we mean a principal submatrix that contains the first k columns and rows.

For a matrix $M \in \mathbb{F}^{n\times n}$, the sparsity $s(M)$ is the number of non-zero entries in M. We say that a matrix $M \in \mathbb{F}^{n\times n}$ of rank k *contains* an *s-sparse column (row) basis*, if M *contains* a column (row) basis (i.e., a set of k linearly independent columns (rows)) with a total of at most s non-zero entries.

▶ **Definition 4** (Minrank [4, 18]). [4] Let $G = (V, A)$ be a graph on $n = |V|$ vertices with the set of directed arcs A. A matrix $M \in \mathbb{F}^{n\times n}$ *represents* G if $M_{i,i} \neq 0$ for every $i \in [n]$, and

[3] Throughout the paper we assume that graphs under consideration do not contain self-loops. In particular, neither G nor $\bar{G}$ has self-loops.

[4] In this paper we consider the directed version of minrank. Since the minrank of a directed graph does not exceed the minrank of its undirected counterpart, a lower bound for a directed random graph implies the same lower bound for an undirected random graph. The bound is tight for both directed and undirected random graphs (see Theorem 12).

$M_{i,j} = 0$ whenever $(i, j) \notin A$ and $i \neq j$. The minrank of G over $\mathbb{F}$ is

$$\mathsf{minrk}_{\mathbb{F}}(G) = \min_{M \text{ represents } G} \mathsf{rk}(M) \ .$$

We say that two graphs *differ at only one vertex* if they differ only in arcs leaving one vertex. Following [13, 14], to amplify the probability in Theorem 10, we shall use the following form of Azuma's inequality for the vertex exposure martingale.

▶ **Lemma 5** (Corollary 7.2.2 and Theorem 7.2.3 in [2]). *Let $f(\cdot)$ be a function that maps directed graphs to $\mathbb{R}$. If f satisfies the inequality $|f(H) - f(H')| \leq 1$ whenever the graphs H and H' differ at only one vertex, then*

$$\Pr[|f(\mathcal{G}_{n,p}) - \mathbb{E}[f(\mathcal{G}_{n,p})]| > \lambda\sqrt{n-1}] < 2e^{-\lambda^2/2} \ .$$

3 The Minrank of a Random Graph

The following elementary linear-algebraic lemma shows that a matrix $M \in \mathbb{F}^{n\times n}$ of rank k is fully specified by k linearly independent rows, k linearly independent columns, and their $2k$ indices. In what follows, we denote by $\mathcal{M}_{n,k}$ the set of matrices from $\mathbb{F}^{n\times n}$ of rank k.

▶ **Lemma 6** (Row and column bases encode the entire matrix). *The mapping $\phi\colon \mathcal{M}_{n,k} \to (\mathbb{F}^{1\times n})^k \times (\mathbb{F}^{n\times 1})^k \times [n]^{2k}$ defined as*

$$\phi(M) = (R, C, i_1, \ldots, i_k, j_1, \ldots, j_k) \ ,$$

is a one-to-one mapping, where $R = (\mathrm{Row}_{i_1}(M), \ldots, \mathrm{Row}_{i_k}(M))$ and $C = (\mathrm{Col}_{j_1}(M), \ldots, \mathrm{Col}_{j_k}(M))$ are, respectively, a row basis and a column basis of $M \in \mathcal{M}_{n,k}$.

Proof. We first claim that the intersection of R and C has full rank, i.e., that the submatrix $M' \in \mathbb{F}^{k\times k}$ obtained by taking rows $i_1, \ldots, i_k$ and columns $j_1, \ldots, j_k$ has rank k. This is a standard fact, see, e.g., [15, p20, Section 0.7.6]. We include a proof for completeness. Assume for convenience that $(i_1, \ldots, i_k) = (1, \ldots, k)$ and $(j_1, \ldots, j_k) = (1, \ldots, k)$. Next, assume towards contradiction that $\mathsf{rk}(M') = \mathsf{rk}(\{\mathrm{Col}_1(M'), \ldots, \mathrm{Col}_k(M')\}) = k' < k$. Since C is a column basis of M, every column $\mathrm{Col}_i(M)$ is a linear combination of vectors from C, and in particular, every $\mathrm{Col}_i(M')$ is a linear combination of $\{\mathrm{Col}_1(M'), \ldots, \mathrm{Col}_k(M')\}$. Therefore, the $k \times n$ submatrix $M'' := (\mathrm{Col}_1^{\leq k}(M), \ldots, \mathrm{Col}_n^{\leq k}(M))$ has rank k'. On the other hand, the k rows of M'': $\mathrm{Row}_1(M), \ldots, \mathrm{Row}_k(M)$ were chosen to be linearly independent by construction. Thus, $\mathsf{rk}(M'') = k > k'$, which leads to a contradiction.

In order to show that ϕ is one-to-one, we show that R and C (together with their indices) uniquely determine the remaining entries of M. We again assume for convenience that $(i_1, \ldots, i_k) = (1, \ldots, k)$ and $(j_1, \ldots, j_k) = (1, \ldots, k)$. Consider any column vector $\mathrm{Col}_i(M)$, $i \in [n] \setminus [k]$. By definition, $\mathrm{Col}_i(M) = \sum_{t=1}^{k} \alpha_{i,t} \cdot \mathrm{Col}_t(M)$ for some coefficient vector $\alpha_i := (\alpha_{i,1}, \ldots, \alpha_{i,k}) \in \mathbb{F}^{k\times 1}$. Thus, in order to completely specify all the entries of $\mathrm{Col}_i(M)$, it suffices to determine the coefficient vector α_i. But M' has full rank, hence the equation

$$M'\alpha_i^T = \mathrm{Col}_i^{\leq k}(M)$$

has a *unique* solution. Therefore, the coefficient vector α_i is fully determined by M' and $\mathrm{Col}_i^{\leq k}(M)$. Thus, the matrix M can be uniquely recovered from R, C and the indices $\{i_1, \ldots, i_k\}, \{j_1, \ldots, j_k\}$. ◀

The following corollary gives us an upper bound on the number of low-rank matrices that contain sparse column and row bases. In what follows, we denote by $\mathcal{M}_{n,k,s}$ the set of matrices over $\mathbb{F}^{n\times n}$ of rank k that contain an s-sparse row basis and an s-sparse column basis.

▶ **Corollary 7** (Efficient encoding of sparse-base matrices)**.**

$$|\mathcal{M}_{n,k,s}| \leq (n \cdot |\mathbb{F}|)^{6s} .$$

Proof. Throughout the proof, we assume without loss of generality that $s \geq k$, as otherwise $|\mathcal{M}_{n,k,s}| = 0$ hence the inequality trivially holds. The function ϕ from Lemma 6 maps matrices from $\mathcal{M}_{n,k,s}$ to $(R, C, i_1, \ldots, i_k, j_1, \ldots, j_k)$, where R and C are s-sparse bases. Therefore, the total number of matrices in $\mathcal{M}_{n,k,s}$ is bounded from above by

$$\left(\binom{kn}{s} \cdot |\mathbb{F}|^s\right)^2 \cdot n^{2k} \leq \left((n^2)^s \cdot |\mathbb{F}|^s\right)^2 \cdot n^{2k} \leq (n \cdot |\mathbb{F}|)^{6s} ,$$

where the last inequality follows from $k \leq s$. ◀

Now we show that a matrix of low rank with nonzero entries on the main diagonal must contain many nonzero entries. To get some intuition on this, notice that a rank 1 matrix with nonzero entries on the diagonal must be nonzero everywhere. Also notice that the assumption on the diagonal is crucial – low rank matrices in general can be very sparse.

▶ **Lemma 8** (Sparsity vs. Rank for matrices with non-zero diagonal)**.** *For any matrix $M \in \mathbb{F}^{n\times n}$ with non-zero entries on the main diagonal (i.e., $M_{i,i} \neq 0$ for all $i \in [n]$), it holds that*

$$s(M) \geq \frac{n^2}{4\mathsf{rk}(M)} .$$

Proof. Let s denote $s(M)$. The average number of nonzero entries in a column of M is s/n. Therefore, Markov's inequality implies that there are at least $n/2$ columns in M *each of which* has sparsity at most $2s/n$. Assume without loss of generality that these are the first $n/2$ columns of M. Now pick a maximal set of linearly independent columns among these columns. We claim that the cardinality of this set is at least $n^2/(4s)$. Indeed, in any set of less than $n^2/(4s)$ columns, the number of coordinates that are nonzero in *at least one* of those columns is less than

$$\frac{n^2}{4s} \cdot \frac{2s}{n} = \frac{n}{2}$$

and therefore there exists a coordinate $i \in \{1, \ldots, n/2\}$ that is zero in all those columns. As a result, the ith column, which by assumption has a nonzero ith coordinate, must be linearly independent of all those columns, in contradiction to the maximality of the set. We therefore get that

$$\mathsf{rk}(M) \geq n^2/(4s) ,$$

as desired. ◀

The last lemma we need is also the least trivial. In order to use Corollary 7, we would like to show that any $n \times n$ matrix of rank k has sparse row and column bases, where by sparse we mean that their sparsity is roughly k/n times that of the entire matrix. If the number of nonzero entries in each row and column was roughly the same, then this would be trivial, as we can take any maximal set of linearly independent columns or rows. However, in general, this might be impossible to achieve. E.g., consider the $n \times n$ matrix whose first k columns are chosen uniformly and the remaining $n - k$ columns are all zero. Then any column basis would have to contain all first k columns (since they are linearly independent with high probability) and hence its sparsity is equal to that of the entire matrix. Instead, what the lemma shows is that one can always choose a *principal submatrix* with the desired property, i.e., that it contains sparse row and column bases, while at the same time having relative rank that is at most that of the original matrix.

▶ **Lemma 9** (Every matrix contains a principal submatrix of low relative-rank and sparse bases). *Let $M \in \mathcal{M}_{n,k}$ be a matrix. There exists a principal submatrix $M' \in \mathcal{M}_{n',k'}$ of M, such that $k'/n' \leq k/n$, and M' contains a column basis and a row basis of sparsity at most*

$$s(M') \cdot \frac{2k'}{n'} .$$

Note that if M contains a zero entry on the main diagonal, the lemma becomes trivial. Indeed, we can take M' to be a 1×1 principal submatrix formed by this zero entry. Thus, the lemma is only interesting for matrices M without zero elements on the main diagonal (i.e., when every principal submatrix has rank greater than 0).

Proof. We prove the statement of the lemma by induction on n. The base case $n = 1$ holds trivially.

Now let $n > 1$, and assume that the statement of the lemma is proven for every $m \times m$ matrix for $1 \leq m < n$. Let $s(i)$ be the number of nonzero entries in the ith column plus the number of non-zero entries in the ith row (note that a nonzero entry on the diagonal is counted twice). Let also $s_{\max} = \max_i s(i)$. By applying the same permutation to the columns and rows of M we can assume that $s(1) \leq s(2) \leq \cdots \leq s(n)$ holds.

If for some $1 \leq n' < n$, the leading principal submatrix M' of dimensions $n' \times n'$ has rank at most $k' \leq n'k/n$, then we use the induction hypothesis for M'. This gives us a principal submatrix M'' of dimensions $n'' \times n''$ and rank k'', such that M'' contains a column basis and a row basis of sparsity at most $s(M'') \cdot \frac{2k''}{n''}$. Also, by induction hypothesis $k''/n'' \leq k'/n' \leq k/n$, which proves the lemma statement in this case.

Now we assume that for all $n' < n$, the rank of the leading principal submatrix of dimension $n' \times n'$ is greater than $n'k/n$. We prove that the lemma statement holds for $M' = M$ for a column basis, and an analogous proof gives the same result for a row basis.

For every $0 \leq i \leq s_{\max}$, let $a_i = |\{j : s(j) = i\}|$. Note that

$$\sum_{i=0}^{s_{\max}} a_i = n . \tag{1}$$

Let us select a column basis of cardinality k by greedily adding linearly independent vectors to the basis in non-decreasing order of $s(i)$. Let k_i be the number of selected vectors j with $s(j) = i$. Then

$$\sum_{i=0}^{s_{\max}} k_i = k. \tag{2}$$

Next, for any $0 \leq t < s_{\max}$, consider the leading principal submatrix given by indices i with $s(i) \leq t$. The rank of this matrix is at most $k' = \sum_{i=0}^{t} k_i$, and its dimensions are $n' \times n'$, where $n' = \sum_{i=0}^{t} a_i < n$. Thus by our assumption $k'/n' \geq k/n$, or equivalently,

$$\sum_{i=0}^{t} k_i \geq \frac{k}{n} \cdot \sum_{i=0}^{t} a_i \ . \tag{3}$$

From (1) and (2),

$$\sum_{i=0}^{s_{\max}} k_i = \frac{k}{n} \cdot \sum_{i=0}^{s_{\max}} a_i \ . \tag{4}$$

Now, (3) and (4) imply that for all $0 \leq t \leq s_{\max}$:

$$\sum_{i=t}^{s_{\max}} k_i \leq \frac{k}{n} \cdot \sum_{i=t}^{s_{\max}} a_i \ . \tag{5}$$

To finish the proof, notice that the sparsity of the constructed basis of M is at most

$$\sum_{i=1}^{s_{\max}} i \cdot k_i = \sum_{t=1}^{s_{\max}} \sum_{i=t}^{s_{\max}} k_i \overset{(5)}{\leq} \frac{k}{n} \cdot \sum_{t=1}^{s_{\max}} \sum_{i=t}^{s_{\max}} a_i = \frac{k}{n} \cdot \sum_{i=1}^{s_{\max}} i \cdot a_i = s(M) \cdot \frac{2k}{n} \ . \qquad ◀$$

Now we are ready to prove our main result – a lower bound on the minrank of a random graph.

▶ **Theorem 10.**

$$\Pr\left[\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p}) \geq \Omega\left(\frac{n\log(1/p)}{\log\left(n|\mathbb{F}|/p\right)}\right)\right] \ \geq\ 1 - e^{-\Omega\left(\frac{n\log^2(1/p)}{\log^2(n|\mathbb{F}|/p)}\right)} \ .$$

Proof. Let us bound from above probability that a random graph $\mathcal{G}_{n,p}$ has minrank at most

$$k := \frac{n\log(1/p)}{C\log\left(n|\mathbb{F}|/p\right)} \ ,$$

for some constant C to be chosen below.

Recall that by Lemma 9, every matrix of rank at most k contains a principal submatrix $M' \in \mathcal{M}_{n',k'}$ of sparsity $s' = s(M')$ with column and row bases of sparsity at most

$$s' \cdot \frac{2k}{n} \ ,$$

where $k'/n' \leq k/n$. By Corollary 7, there are at most $(n' \cdot |\mathbb{F}|)^{6(2s'k/n)}$ such matrices M', and (for any s') there are $\binom{n}{n'}$ ways to choose a principal submatrix of size n' in a matrix of size $n \times n$. Furthermore, recall that Lemma 8 asserts that for every n', k',

$$s' \geq \frac{n'^2}{4k'}. \tag{6}$$

Finally, since M' contains at least $s' - n'$ off-diagonal non-zero entries, $\mathcal{G}_{n,p}$ contains it with probability at most $p^{s'-n'}$. We therefore have

$$\begin{aligned}
&\Pr\left[\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p}) \leq k\right] \\
&\leq \sum_{k',n',s'} \Pr\left[\mathcal{G}_{n,p} \text{ contains } M' \in \mathcal{M}_{n',k'},\ s(M') = s', s(\text{bases of } M') \leq s' \cdot \frac{2k}{n}\right] \\
&\leq \sum_{k',n',s'} \binom{n}{n'} \cdot p^{s'-n'} \cdot (n' \cdot |\mathbb{F}|)^{12s'k/n} \\
&\leq \sum_{k',n',s'} 2^{n'\log n - s'\log(1/p) + n'\log(1/p) + (12s'k/n)\log(n'|\mathbb{F}|)} \ ,
\end{aligned} \tag{7}$$

where all the summations are taken over n', k', s.t. $k'/n' \le k/n$ and $s' \ge \frac{n'^2}{4k'}$, and the first inequality is again by Lemma 9. We now argue that for sufficiently large constant C, all positive terms in the exponent of (7) are dominated by the magnitude of the negative term $(s' \log(1/p))$. Indeed:

$$\begin{aligned} n' \log n + n' \log(1/p) + (12s'k/n)\log(n'|\mathbb{F}|) &= n' \log(n/p) + (12s'k/n)\log(n'|\mathbb{F}|) \\ \le (4s'k'/n')\log(n/p) + (12s'k/n)\log(n|\mathbb{F}|) &\le (16s'k/n)\log(n|\mathbb{F}|/p) \\ &= (16s'/C)\log(1/p)\ , \end{aligned}$$

where the first inequality follows from (6), and the second one follows from $k'/n' \le k/n$.

Thus, for $C > 16$,

$$\Pr\left[\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p}) \le \frac{n\log(1/p)}{C\log(n|\mathbb{F}|/p)}\right] \le n^4 \cdot 2^{-\Omega(s'\log(1/p))} \le 2^{-\Omega(\log(n))},$$

where the last inequality follows from:

$$s'\log(1/p) \ge \frac{n'^2\log(1/p)}{4k'} \ge \frac{n\log(1/p)}{4k} = \frac{n\log(1/p)C\log(n|\mathbb{F}|/p)}{4n\log(1/p)} \ge \frac{C\log n}{4}\ .$$

In particular, $\mathbb{E}[\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p})] \ge \frac{n\log(1/p)}{2C\log(n|\mathbb{F}|/p)}$. Furthermore, note that changing a single row (or column) of a matrix can change its minrank by at most 1, hence the minrank of two graphs that differ in one vertex differs by at most 1. We may thus apply Lemma 5 with $\lambda = \Theta\left(\frac{\sqrt{n}\log(1/p)}{\log(n|\mathbb{F}|/p)}\right)$ to obtain

$$\Pr\left[\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p}) \ge \Omega\left(\frac{n\log(1/p)}{\log(n|\mathbb{F}|/p)}\right)\right] \ \ge\ 1 - e^{-\Omega\left(\frac{n\log^2(1/p)}{\log^2(n|\mathbb{F}|/p)}\right)}\ .$$

as desired. ◀

▶ **Corollary 11.** *For a constant $0 < p < 1$ and a field $\mathbb{F}$ of size $|\mathbb{F}| < n^{O(1)}$,*

$$\Pr[\,\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p}) \ge \Omega(n/\log n)\,] \ \ge\ 1 - e^{-\Omega\left(n/\log^2 n\right)}\ .$$

3.1 Tightness of Theorem 10

In this section, we show that Theorem 10 provides a tight bound for all values of p bounded away from 1 (i.e., $p \le 1 - \Omega(1)$). (See also the end of the section for the regime of p close to 1.)

▶ **Theorem 12.** *For any p bounded away from 1,*

$$\Pr\left[\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p}) = O\left(\frac{n\log(1/p)}{\log n + \log(1/p)}\right)\right] \ \ge\ 1 - e^{-\Omega(n)}\ .$$

Proof. We can assume that $p > n^{-1/8}$ as otherwise the statement is trivial.

As we saw in the introduction, in the case of a clique (a graph with an arc between every pair of distinct vertices) it is enough to broadcast only one bit. This simple observation leads to the "clique-covering" upper bound: If a directed graph G can be covered by m cliques, then $\mathsf{minrk}_{\mathbb{F}}(G) \le m$ [11, 4, 14]. Note that the minimal number of cliques needed to cover G is exactly $\chi(\bar{G})$. Thus, we have the following upper bound: For any field $\mathbb{F}$ and any directed graph G,

$$\mathsf{minrk}_{\mathbb{F}}(G) \le \chi(\bar{G})\ . \tag{8}$$

Since the complement of $\mathcal{G}_{n,p}$ is $\mathcal{G}_{n,1-p}$, it follows from (8) that an upper bound on $\chi(\mathcal{G}_{n,1-p})$ implies an upper bound on $\mathsf{minrk}_{\mathbb{F}}(\mathcal{G}_{n,p})$.

Let $\mathcal{G}^{-}_{n,p}$ denote a random Erdős-Rényi *undirected* graph on n vertices, where each edge is drawn independently with probability p. For constant p, the classical result of Bollobás [7] asserts that the chromatic number of an undirected random graph satisfies

$$\Pr\left[\chi(\mathcal{G}^{-}_{n,1-p}) \leq \frac{n\log(1/p)}{2\log n}(1+o(1))\right] > 1 - e^{-\Omega(n)} \; . \tag{9}$$

In fact, Pudlák, Rödl, and Sgall [21] showed that (9) holds for any $p > n^{-1/4}$.

Since we define the chromatic number of a directed graph to be the chromatic number of its undirected counterpart, $\chi(\mathcal{G}_{n,1-p}) = \chi(\mathcal{G}^{-}_{n,1-p^2})$. The bound (9) depends on p only logarithmically $(\log(1/p))$, thus, asymptotically the same bounds hold for the chromatic number of a random directed graph. ◀

The lower bound of Theorem 10 is also almost tight for the other extreme regime of $p = 1-\varepsilon$, where $\varepsilon = o(1)$. Łuczak [19] proved that for $p = 1 - \Omega(1/n)$,

$$\Pr\left[\chi(\mathcal{G}^{-}_{n,1-p}) \leq \frac{n(1-p)}{2\log n(1-p)}(1+o(1))\right] > 1 - (n(1-p))^{-\Omega(1)} \; . \tag{10}$$

When $p = 1-\varepsilon$, the upper bound (10) matches the lower bound of Theorem 10 for $\varepsilon \geq n^{-1+\Omega(1)}$. For $\varepsilon = O(n^{-1})$, (10) gives an asymptotically tight upper bound of $O(1)$. Thus, we only have a gap between the lower bound of Theorem 10 and known upper bounds when $p = 1-\varepsilon$ and $\omega(1) \leq n\varepsilon \leq n^{o(1)}$.

Acknowledgements. We would like to thank Ishay Haviv for his valuable comments.

References

1 Rudolf Ahlswede, Ning Cai, Shuo-Yen Robert Li, and Raymond W. Yeung. Network information flow. *IEEE Trans. Inf. Theory*, 46(4):1204–1216, 2000.

2 Noga Alon and Joel H. Spencer. *The Probabilistic Method.* Wiley Series in Discrete Mathematics and Optimization. Wiley, 2016.

3 Fatemeh Arbabjolfaei and Young-Han Kim. Three stories on a two-sided coin: Index coding, locally recoverable distributed storage, and guessing games on graphs. In *Allerton Conf. Control, Communication and Computing 2015*, pages 843–850. IEEE, 2015.

4 Ziv Bar-Yossef, Yitzhak Birk, T. S. Jayram, and Tomer Kol. Index coding with side information. In *FOCS 2006*, pages 197–206. IEEE, 2006.

5 Yitzhak Birk and Tomer Kol. Informed-source coding-on-demand (ISCOD) over broadcast channels. In *INFOCOM 1998*, pages 1257–1264, 1998.

6 Yitzhak Birk and Tomer Kol. Coding on demand by an informed source (ISCOD) for efficient broadcast of different supplemental data to caching clients. *IEEE Trans. Information Theory*, 52(6):2825–2830, 2006. doi:10.1109/TIT.2006.874540.

7 Béla Bollobás. The chromatic number of random graphs. *Combinatorica*, 8(1):49–55, 1988.

8 Chris Calabro. A lower bound on the size of series-parallel graphs dense in long paths, 2008. ECCC, TR08-110.

9 Eden Chlamtac and Ishay Haviv. Linear index coding via semidefinite programming. *Combinatorics, Probability & Computing*, 23(2):223–247, 2014. doi:10.1017/S0963548313000564.

10 Michelle Effros, Salim Y. El Rouayheb, and Michael Langberg. An equivalence between network coding and index coding. *IEEE Trans. Information Theory*, 61(5):2478–2487, 2015. doi:10.1109/TIT.2015.2414926.

11 Willem Haemers. An upper bound for the Shannon capacity of a graph. In *Colloq. Math. Soc. János Bolyai*, volume 25, pages 267–272, 1978.

12 Willem Haemers. On some problems of Lovász concerning the Shannon capacity of a graph. *IEEE Trans. Inf. Theory*, 25(2):231–232, 1979.

13 H. Tracy Hall, Leslie Hogben, Ryan Martin, and Bryan Shader. Expected values of parameters associated with the minimum rank of a graph. *Linear Algebra and its Applications*, 433(1):101–117, 2010.

14 Ishay Haviv and Michael Langberg. On linear index coding for random graphs. In *ISIT 2012*, pages 2231–2235. IEEE, 2012.

15 Roger A. Horn and Charles R. Johnson. *Matrix analysis*. Cambridge University Press, Cambridge, second edition, 2013.

16 Stasys Jukna and Georg Schnitger. Min-rank conjecture for log-depth circuits. *J. Comput. Syst. Sci.*, 77(6):1023–1038, 2011. doi:10.1016/j.jcss.2009.09.003.

17 Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. In *STOC 1995*, pages 596–605, New York, NY, USA, 1995. ACM. doi:10.1145/225058.225277.

18 Eyal Lubetzky and Uri Stav. Non-linear index coding outperforming the linear optimum. In *FOCS 2007*, pages 161–168. IEEE, 2007.

19 Tomasz Łuczak. The chromatic number of random graphs. *Combinatorica*, 11(1):45–54, 1991.

20 Mihai Patrascu. Towards polynomial lower bounds for dynamic problems. In *STOC 2010*, pages 603–610, 2010. doi:10.1145/1806689.1806772.

21 Pavel Pudlák, Vojtech Rödl, and Jirí Sgall. Boolean circuits, tensor ranks, and communication complexity. *SIAM J. Comput.*, 26(3):605–633, 1997.

22 Søren Riis. Information flows, graphs and their guessing numbers. *Electr. J. Comb.*, 14(1), 2007. URL: http://www.combinatorics.org/Volume_14/Abstracts/v14i1r44.html.

23 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *MFCS 1977*, pages 162–176, 1977.

24 Leslie G. Valiant. Exponential lower bounds for restricted monotone circuits. In *STOC 1983*, pages 110–117. ACM, 1983.

25 Leslie G. Valiant. Why is Boolean complexity theory difficult. *Boolean Function Complexity*, 169:84–94, 1992.

26 Emanuele Viola. On the power of small-depth computation. *Foundations and Trends in Theoretical Computer Science*, 5(1):1–72, 2009.

27 Raymond W. Yeung and Zhen Zhang. Distributed source coding for satellite communications. *IEEE Trans. Inf. Theory*, 45(4):1111–1120, 1999. doi:10.1109/18.761254.

Efficiently Decodable Codes for the Binary Deletion Channel*

Venkatesan Guruswami[1] and Ray Li[2]

1 **Carnegie Mellon University, Pittsburgh, PA**
`venkatg@cs.cmu.edu`

2 **Carnegie Mellon University, Pittsburgh, PA**
`ryli@andrew.cmu.edu`

Abstract

In the random deletion channel, each bit is deleted independently with probability p. For the random deletion channel, the *existence* of codes of rate $(1-p)/9$, and thus bounded away from 0 for any $p < 1$, has been known. We give an explicit construction with polynomial time encoding and deletion correction algorithms with rate $c_0(1-p)$ for an absolute constant $c_0 > 0$.

1998 ACM Subject Classification E.4 Coding and Information Theory

Keywords and phrases Coding theory, Combinatorics, Synchronization errors, Channel capacity

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.47

1 Introduction

We consider the problem of designing error-correcting codes for reliable and efficient communication on the binary deletion channel. The *binary deletion channel* (BDC) *deletes* each transmitted bit independently with probability p, for some $p \in (0, 1)$ which we call the *deletion probability*. Crucially, the location of the deleted bits are *not* known at the decoder, who receives a *subsequence* of the original transmitted sequence. The loss of synchronization in symbol locations makes the noise model of deletions challenging to cope with. As one indication of this, we still do not know the channel capacity of the binary deletion channel. Quoting from the first page of Mitzenmacher's survey [17]: "Currently, we have no closed-form expression for the capacity, nor do we have an efficient algorithmic means to numerically compute this capacity." This is in sharp contrast with the noise model of bit erasures, where each bit is independently replaced by a '?' with probability p (the binary erasure channel (BEC)), or of bit errors, where each bit is flipped independently with probability p (the binary symmetric channel (BSC)). The capacity of the BEC and BSC equal $1-p$ and $1-h(p)$ respectively, and we know codes of polynomial complexity with rate approaching the capacity in each case.

The capacity of the binary deletion channel is clearly at most $1-p$, the capacity of the simpler binary erasure channel. Diggavi and Grossglauser [3] establish that the capacity of the deletion channel for $p \leq \frac{1}{2}$ is at least $1-h(p)$. Kalai, Mitzenmacher, and Sudan [11] proved this lower bound is tight as $p \to 0$, and Kanoria and Montanari [12] determined a series expansion that can be used to determine the capacity exactly. Turning to large p, Rahmati and Duman [18] prove that the capacity is at most $0.4143(1-p)$ for $p \geq 0.65$. Drinea and Mitzenmacher [4, 5] proved that the capacity of the BDC is at least $(1-p)/9$,

* Research supported in part by NSF grant CCF-1422045.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 47; pp. 47:1–47:13
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

which is within a constant factor of the upper bound. In particular, the capacity is positive for every $p < 1$, which is perhaps surprising. The asymptotic behavior of the capacity of the BDC at both extremes of $p \to 0$ and $p \to 1$ is thus known.

This work is concerned with *constructive* results for coding for the binary deletion channel. That is, we seek codes that can be constructed, encoded, and decoded from deletions caused by the BDC, in polynomial time. Recently, there has been good progress on codes for *adversarial* deletions, including constructive results. Here the model is that the channel can delete an arbitrary subset of pn bits in the n-bit codeword. A code capable of correcting pn worst-case deletions can clearly also correct deletions caused by a BDC with deletion probability $(p - \epsilon)$ with high probability, so one can infer results for the BDC from some results for worst-case deletions. For small p, Guruswami and Wang [9] constructed binary codes of rate $1 - O(\sqrt{p})$ to efficiently correct a p fraction worst-case deletions. So this also gives codes of rate approaching 1 for the BDC when $p \to 0$. For larger p, Kash et al. [13] proved that randomly chosen codes of small enough rate $R > 0$ can correctly decode against pn adversarial deletions when $p \leq 0.17$. Even non-constructively, this remained the best achievability result in terms of correctable deletion fraction until the recent work of Bukh, Guruswami, and Håstad [2] who constructed codes of positive rate efficiently decodable against pn adversarial deletions for any $p < \sqrt{2} - 1$. For adversarial deletions, it is impossible to correct a deletion fraction of $1/2$, whereas the capacity of the BDC is positive for all $p < 1$. So solving the problem for the much harder worst-case deletions is not a viable approach to construct positive rate codes for the BDC for $p > 1/2$.

To the best of our knowledge, explicit efficiently decodable code constructions were not available for the binary deletion channel for arbitrary $p < 1$. We present such a construction in this work. Our rate is worse than the $(1 - p)/9$ achieved non-constructively, but has asymptotically the same dependence on p for $p \to 1$.

▶ **Theorem 1.** *Let $p \in (0, 1)$. There is an explicit a family of binary codes that (1) has rate $(1 - p)/110$, (2) is constructible in polynomial time, (3) encodable in time $O(N)$, and (3) decodable with high probability on the binary deletion channel with deletion probability p in time $O(N^2)$. (Here N is the block length of the code)*

1.1 Some other related work

One work that considers efficient recovery against random deletions is by Yazdi and Dolecek [20]. In their setting, two parties Alice and Bob are connected by a two-way communication channel. Alice has a string X, Bob has string Y obtained by passing X through a binary deletion channel with deletion probability $p \ll 1$, and Bob must recover X. They produce a polynomial-time synchronization scheme that transmits a total of $O(pn \log(1/p))$ bits and allows Bob to recover X with probability exponentially approaching 1.

For other models of random synchronization errors, Kirsch and Drinea [14] prove information capacity lower bounds for channels with i.i.d deletions and duplications. Fertonani et al. [6] prove capacity bounds for binary channels with i.i.d insertions, deletions, and substitutions.

For deletion channels over non-binary alphabets, Rahmati and Duman [18] prove a capacity upper bound of $C_2(p) + (1 - p) \log(|\Sigma|/2)$, where $C_2(p)$ denotes the capacity of the binary deletion channel with deletion probability p, when the alphabet size $|\Sigma|$ is even. In particular, using the best known bound for $C_2(p)$ of $C_2(p) \leq 0.4143(1 - p)$, the upper bound is $(1 - p)(\log |\Sigma| - 0.5857)$.

In [8], the authors of this paper consider the model of *oblivious* deletions, which is in between the BDC and adversarial deletions in power. Here, the channel can delete any pn

bits of the codeword, but must do so without knowledge of the codeword. In this model, they prove the *existence* of codes of positive rate for correcting any fraction $p < 1$ of oblivious deletions.

1.2 Our construction approach

Our construction concatenates a high rate outer code over a large alphabet that is efficiently decodable against a small fraction of *adversarial* insertions and deletions, with a good inner binary code. For the outer code, we can use the recent construction of [10]. To construct the inner code, we first choose a binary code correcting a small fraction of adversarial deletions. By concentration bounds, duplicating bits of a codeword in a disciplined manner is effective against the random deletion channel, so we, for some constant B, duplicate every bit of the binary code $B/(1-p)$ times. We further ensure our initial binary code has only runs of length 1 and 2 to maximize the effectiveness of duplication. We add small buffers of 0s between inner codewords to facilitate decoding.

One might wonder whether it would be possible to use Drinea and Mitzenmacher's existential result [4, 5] of a $(1-p)/9$ capacity lower bound as a black box inner code to achieve a better rate together with efficient decodability. We discuss this approach in §3.2 and elaborate on what makes such a construction difficult to implement.

2 Preliminaries

General Notation. Throughout the paper, $\log x$ refers to the base-2 logarithm. We use interval notation $[a, b] = \{a, a+1, \dots, b\}$ to denote intervals of integers, and we use $[a] = [1, a] = \{1, 2, \dots, a\}$. Let $\text{Binomial}(n, p)$ denote the Binomial distribution.

Words. A *word* is a sequence of symbols from some *alphabet.* We denote explicit words using angle brackets, like $\langle 01011 \rangle$. We denote string concatenation of two words w and w' with ww'. We denote $w^k = ww \cdots w$ where there are k concatenated copies of w.

A *subsequence* of a word w is a word obtained by removing some (possibly none) of the symbols in w.

Let $\Delta_{i/d}(w_1, w_2)$ denote the *insertion/deletion distance* between w_1 and w_2, i.e. the minimum number of insertions and deletions needed to transform w_1 into w_2. By a lemma due to Levenshtein [15], this is equal to $|w_1| + |w_2| - 2\,\text{LCS}(w_1, w_2)$, where LCS denotes the length of the longest common subsequence.

Define a *run* of a word w to be a maximal single-symbol subword. That is, a subword w' in w consisting of a single symbol such that any longer subword containing w' has at least two different symbols. Note the runs of a word partition the word. For example, 110001 has 3 runs: one run of 0s and two runs of 1s.

We say that $c \in \{0, 1\}^m$ and $c' \in \{0, 1\}^m$ are *confusable under δm deletions* if it is possible to apply δm deletions to c and c' and obtain the same result. If δ is understood, we simply say c and c' are *confusable.*

Concentration Bounds. We use the following forms of Chernoff bound.

▶ **Lemma 2** (Chernoff). *Let $A_1, \dots, A_n$ be i.i.d random variables taking values in $[0, 1]$. Let $A = \sum_{i=1}^n A_i$ and $\delta \in [0, 1]$. Then*

$$\mathbf{Pr}[A \le (1-\delta)\,\mathbb{E}[A]] \;\le\; \exp\left(-\delta^2\,\mathbb{E}[A]/2\right) \tag{1}$$

Furthermore,

$$\mathbf{Pr}[A \geq (1+\delta)\,\mathbb{E}[A]] \;\leq\; \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mathbb{E}[A]}. \tag{2}$$

We also have the following corollary, whose proof is in Appendix A.

▶ **Lemma 3.** *Let $0 < \alpha < \beta$. Let $A_1, \ldots, A_n$ be independent random variables taking values in $[0, \beta]$ such that, for all i, $\mathbb{E}[A_i] \leq \alpha$. For $\gamma \in [\alpha, 2\alpha]$, we have*

$$\mathbf{Pr}\left[\sum_{i=1}^{n} A_i \geq n\gamma\right] \leq \exp\left(-\frac{(\gamma-\alpha)^2 n}{3\alpha\beta}\right). \tag{3}$$

3 Efficiently decodable codes for random deletions with p approaching 1

3.1 Construction

We present a family of constant rate codes that decodes with high probability on a binary deletion channel with deletion fraction p (BDC$_p$). These codes have rate $c_0(1-p)$ for an absolute positive constant c_0, which is within a constant of the upper bound $(1-p)$, which even holds for the erasure channel. By Drinea and Mitzenmacher [4] the maximum known rate of a non-efficiently correctable binary deletion channel code is $(1-p)/9$.

The construction is based on the intuition that deterministic codes are better than random codes for the deletion channel. Indeed, for adversarial deletions, length n random codes correct at most $0.22n$ deletions [13], while explicitly constructed codes can correct close to $(\sqrt{2}-1)n$ deletions [2].

We begin by borrowing a result from [9].

▶ **Lemma 4** (Corollary of Lemma 2.3 of [9])**.** *Let $0 < \delta < \frac{1}{2}$. For every binary string $c \in \{0,1\}^m$, there are at most $\delta m\binom{m}{(1-\delta)m}^2$ strings $c' \in \{0,1\}^m$ such that c and c' are confusable under δm deletions.*

The next lemma gives codes against a small fraction of adversarial deletions with an additional run-length constraint on the codewords.

▶ **Lemma 5.** *Let $\delta > 0$. There exists a length m binary code of rate $\mathcal{R} = 0.6942 - 2h(\delta) - O(\log(\delta m)/m)$ correcting a δ fraction of adversarial insertions and deletions such that each codeword contains only runs of size 1 and 2. Furthermore this code is constructible in time $\tilde{O}(2^{(0.6942+\mathcal{R})m})$.*

Proof. It is easy to show that the number of codewords with only runs of 1 and 2 is F_m, the mth Fibonacci number, and it is well known that $F_m = \varphi^m + o(1) \approx 2^{0.6942m}$ where φ is the golden ratio. Now we construct the code by choosing it greedily. Each codeword is confusable with at most $\delta m\binom{m}{(1-\delta)m}^2$ other codewords, so the number of codewords we can choose is at least

$$\frac{2^{0.6942m}}{\delta m\binom{m}{(1-\delta)m}^2} \;=\; 2^{m(0.6942-2h(\delta)-O(\log(\delta m)/m))}. \tag{4}$$

We can find all words of length m whose run lengths are only 1 and 2 by recursion in time $O(F_m) = O(2^{0.6942m})$. Running the greedy algorithm, we need to, for at most $F_m \cdot 2^{\mathcal{R}m}$ pairs of such words, determine whether the pair is confusable (we only need to check confusability of a candidate word with words already added to the code). Checking confusability of two words under adversarial deletions reduces to checking whether the longest common subsequence is at least $(1-\delta)m$, which can be done in time $O(m^2)$. This gives an overall runtime of $O(m^2 \cdot F_m \cdot 2^{\mathcal{R}m}) = \tilde{O}(2^{(0.6942+\mathcal{R})m})$. ◀

▶ **Corollary 6.** *There exists a constant m_0^* such that for all $m \geq m_0^*$, there exists a length m binary code of rate $\mathcal{R}_{in} = 0.555$ correcting a $\delta_{in} = 0.0083$ fraction of adversarial insertions and deletions such that each codeword contains runs of size 1 and 2 only and each codeword starts and ends with a 1. Furthermore this code is constructible in time $O(2^{1.25m})$.*

Our construction utilizes the following result as a black box for efficiently coding against an arbitrary fraction of insertions and deletions with rate approaching capacity.

▶ **Theorem 7** (Theorem 1.1 of [10]). *For any $0 \leq \delta < 1$ and $\epsilon > 0$, there exists a code C over alphabet Σ, with $|\Sigma| = \mathrm{poly}(1/\epsilon)$, with block length n, rate $1-\delta-\epsilon$, and is efficiently decodable from δn insertions and deletions. The code can be constructed in time $\mathrm{poly}(n)$, encoded in time $O(n)$, and decoded in time $O(n^2)$.*

We apply Theorem 7 for small δ, so we also could use the high rate binary code construction of [7] as an outer code.

We now turn to our code construction for Theorem 1.

The code. Let $B = 60, B^* = 1.4\bar{3}B = 86, \eta = \frac{1}{1000}, \delta_{out} = \frac{1}{1000}$. Let

$$m_0 = \max(\alpha \log(1/\delta_{out})/\eta, m_0^*)\,,$$

where α is a sufficiently large constant and where m_0^* is given by Corollary 6. Let $\epsilon_{out} > 0$ be small enough such that the alphabet Σ, given by Theorem 7 with $\epsilon = \epsilon_{out}$ and $\delta = \delta_{out}$, satisfies $|\Sigma| \geq m_0$, and let C_{out} be the corresponding code.

Let $C_{in} : |\Sigma| \to \{0,1\}^m$ be the code given by Corollary 6, and let $\mathcal{R}_{in} = 0.555, \delta_{in} = 0.0083$, and $m = \frac{1}{\mathcal{R}_{in}} \log|\Sigma| = O(\log(1/\epsilon))$ be the rate, tolerable deletion fraction, and block length of the code, respectively ($\mathcal{R}_{in}$ and δ_{in} are given by Corollary 6). Each codeword of C_{in} has runs of length 1 and 2 only, and each codeword starts and ends with a 1. This code is constructed greedily.

Our code is a modified concatenated code. We encode our message as follows.

- *Outer Code.* First, encode the message into the outer code, C_{out}, to obtain a word $c^{(out)} = \sigma_1 \dots \sigma_n$.
- *Concatenation with Inner Code.* Encode each outer codeword symbol $\sigma_i \in \Sigma$ by the inner code C_{in}.
- *Buffer.* Insert a buffer of ηm 0s between adjacent inner codewords. Let the resulting word be $c^{(cat)}$. Let $c_i^{(in)} = C_{in}(\sigma_i)$ denote the encoded inner codewords of $c^{(cat)}$.
- *Duplication.* After concatenating the codes and inserting the buffers, replace each character (including characters in the buffers) with $\lceil B/(1-p) \rceil$ copies of itself to obtain a word of length $N := Bnm/(1-p)$. Let the resulting word be c, and the corresponding inner codewords be $\{c_i^{(dup)}\}$.

Rate. The rate of the outer code is $1-\delta_{out}-\epsilon_{out}$, the rate of the inner code is $\mathcal{R}_{in}$, the buffer and duplications multiply the rate by $\frac{1}{1+\eta}$ and $(1-p)/B$ respectively. This gives a total rate that is slightly greater than $(1-p)/110$.

Notation. Let s denote the received word after the codeword c is passed through the deletion channel. Note that (i) every bit of c can be identified with a bit in $c^{(cat)}$, and (ii) each bit in the received word s can be identified with a bit in c. Thus, we can define relations $f^{(dup)} : c^{(cat)} \to c$, and $f^{(del)} : c \to s$ (that is, relations on the indices of the strings). These are not functions because some bits may be mapped to multiple (for $f^{(dup)}$) or zero (for $f^{(del)}$) bits. Specifically, $f^{(del)}$ and $f^{(dup)}$ are the inverses of total functions. In this way, composing these relations (i.e. composing their inverse functions) if necessary, we can speak about the *image* and *pre-image* of bits or subwords of one of $c^{(cat)}, c$, and s under these relations. For example, during the Duplication step of encoding, a bit $\langle b_j \rangle$ of $c^{(cat)}$ is replaced with $B/(1-p)$ copies of itself, so the corresponding string $\langle b_j \rangle^{B/(1-p)}$ in c forms the *image* of $\langle b_j \rangle$ under $f^{(dup)}$, and conversely the *pre-image* of the duplicated string $\langle b_j \rangle^{B/(1-p)}$ is that bit $\langle b_j \rangle$.

Decoding algorithm

- *Decoding Buffer.* First identify all runs of 0s in the received word with length at least $B\eta m/2$. These are our *decoding buffers* that divide the word into *decoding windows*, which we identify with subwords of s.
- *Deduplication.* Divide each decoding window into runs. For each run, if it has strictly more than B^* copies of a bit, replace it with as two copies of that bit, otherwise replace it with one copy. For example, $\langle 0 \rangle^{2B}$ gets replaced with $\langle 00 \rangle$ while $\langle 0 \rangle^{B}$ gets replaced with $\langle 0 \rangle$. For each decoding window, concatenate these runs of length 1 and 2 in their original order in the decoding window to produce a *deduplicated* decoding window.
- *Inner Decoding.* For each deduplicated decoding window, decode an outer symbol $\sigma \in \Sigma_{out}$ from each decoding window by running the brute force deletion correction algorithm for C_{in}. That is, for each deduplicated decoding window $s_*^{(in)}$, find by brute force a codeword $c_*^{(in)}$ in C_{in} that such that $\Delta_{i/d}(c_*^{(in)}, s_*^{(in)}) \leq \delta_{in} m$. If $c_*^{(in)}$ is not unique or does not exist, do not decode an outer symbol σ from this decoding window. Concatenate the decoded symbols σ in the order in which their corresponding decoding windows appear in the received word s to obtain a word $s^{(out)}$.
- *Outer Decoding.* Decode the message $\mathfrak{m}$ from $s^{(out)}$ using the decoding algorithm of C_{out} in Theorem 7.

For purposes of analysis, label as $s_i^{(dup)}$ the decoding window whose pre-image under $f^{(del)}$ contains indices in $c_i^{(dup)}$. If this decoding window is not unique (that is, the image of $c_i^{(dup)}$ contains bits in multiple decoding windows), then assign $s_i^{(dup)}$ arbitrarily. Note this labeling may mean some decoding windows are unlabeled, and also that some decoding windows may have multiple labels. In our analysis, we show both occurrences are rare. For a decoding window $s_i^{(dup)}$, denote the result of $s_i^{(dup)}$ after Deduplication to be $s_i^{(in)}$.

The following diagram depicts the encoding and decoding steps.

The pair $(\{c_i^{(in)}\}_i, c^{(cat)})$ indicates that, at that step of encoding, we have produced the word $c^{(cat)}$, and the sequence $\{c_i^{(in)}\}_i$ are the "inner codewords" of $c^{(cat)}$ (that is, the words in between what would be identified by the decoder as decoding buffers). The pair $(\{c_i^{(dup)}\}_i, c)$ is used similarly.

$$\begin{tikzcd}
\mathfrak{m} \arrow[r, "C_{out}"] & c^{(out)} \arrow[r, "C_{in}{,}Buf"] & \left(\left\{c_i^{(in)}\right\}_i, c^{(cat)}\right) \arrow[r, "Dup"] & \left(\left\{c_i^{(dup)}\right\}_i, c\right) \arrow[dl, bend left, "\text{BDC}"] & & & \\
& & s \arrow[r, "DeBuf"] & \left\{s_i^{(dup)}\right\}_i \arrow[r, "DeDup"] & \left\{s_i^{(in)}\right\}_i \arrow[r, "\mathrm{Dec}_{in}"] & s^{(out)} \arrow[r, "\mathrm{Dec}_{out}"] & \mathfrak{m}
\end{tikzcd}$$

Runtime. The outer code is constructible in poly(n) time and the inner code is constructible in time $O(2^{1.25m}) = \text{poly}(1/\epsilon)$, which is a constant, so the total construction time is poly(N).

Encoding in the outer code is linear time, each of the n inner encodings is constant time, and adding the buffers and applying duplications each can be done in linear time. The overall encoding time is thus $O(N)$.

The Buffer step of the decoding takes linear time. The Deduplication step of each inner codeword takes constant time, so the entire step takes linear time. For each inner codeword, Inner Decoding takes time $O(m^2 2^m) = \text{poly}(1/\epsilon)$ by brute force search over the 2^m possible codewords: checking each of the 2^m codewords is a longest common subsequence computation and thus takes time $O(m^2)$, giving a total decoding time of $O(m^2 2^m)$ for each inner codeword. We need to run this inner decoding $O(n)$ times, so the entire Inner Decoding step takes linear time. The Outer Decoding step takes $O(n^2)$ time by Theorem 7. Thus the total decoding time is $O(N^2)$.

Correctness. Note that, if an inner codeword is decoded incorrectly, then one of the following holds.

1. *Spurious Buffer.* A spurious decoding buffer is identified in the corrupted codeword during the Buffer step.
2. *Deleted Buffer.* A decoding buffer neighboring the codeword is deleted.
3. *Inner Decoding Failure.* Running the Deduplication and Inner Decoding steps on $s_i^{(dup)}$ computes the inner codeword incorrectly.

We show that, with high probability, the number of occurrences of each of these events is small.

The last case is the most nontrivial, so we deal with it first, assuming the codeword contains no spurious decoding buffers and the neighboring decoding buffers are not deleted. In particular, we consider an i such that our decoding window $s_i^{(dup)}$ whose pre-image under $f^{(del)}$ only contains bits in $c_i^{(dup)}$ (because no deleted buffer) and no bits in the image of $c_i^{(dup)}$ appear in any other decoding window (because no spurious buffer).

Recall that the inner code C_{in} can correct against $\delta_{in} = 0.0083$ fraction of adversarial insertions and deletions. Suppose an inner codeword $c_i^{(in)} = r_1 \dots r_k \in C_{in}$ has k runs r_j each of length 1 or 2, so that $m/2 \le k \le m$.

▶ **Definition 8.** A subword of α identical bits in the received word s is
- *type-0* if $\alpha = 0$,
- *type-1* if $\alpha \in [1, B^*]$,
- *type-2* if $\alpha \in [B^* + 1, \infty)$.

By abuse of notation, we say that a length 1 or 2 run r_j of the inner codeword $c_i^{(in)}$ has *type-t_j* if the image of r_j in s under $f^{(del)} \circ f^{(dup)}$ forms a type-t_j subword.

Let $t_1, \dots, t_k$ be the types of the runs $r_1, \dots, r_k$, respectively. The image of a run r_j under $f^{(del)} \circ f^{(dup)}$ has length distributed as Binomial$(B|r_j|/(1-p), 1-p)$. Let $\delta = 0.4\overline{3}$ be such that $B^* = (1+\delta)B$. By the Chernoff bounds in Lemma 2, the probability that a run r_j of length 1 is type-2 is

$$\Pr_{Z \sim \text{Binomial}(B/(1-p),1-p)}[Z > B^*] < \left(e^{\delta}/(1+\delta)^{1+\delta}\right)^B < 0.0071. \tag{5}$$

Similarly, the probability that a run r_j of length-2 is type-1 is at most

$$\Pr_{Z \sim \text{Binomial}(2B/(1-p),1-p)}[Z \le B^*] < e^{-((1-\delta)/2)^2 B} < 0.0081. \tag{6}$$

The probability any run is type-0 is at most $\mathbf{Pr}_{Z \sim \text{Binomial}(B/(1-p),1-p)}[Z = 0] < e^{-B} < 10^{-10}$.

We now have established that, for runs r_j in $c_i^{(in)}$, the probability that the number of bits in the image of r_j in s under $f^{(del)} \circ f^{(dup)}$ is "incorrect" (between 1 and B^* for length 2 runs, and greater than B^* for length 1 runs), is at most 0.0081, which is less than δ_{in}. If the only kinds of errors in the Local Decoding step were runs of c of length 1 becoming runs of length 2 and runs of length 2 become runs of length 1, then we have that, by concentration bounds, with probability $1 - 2^{-\Omega(m)}$, the number of insertions deletions needed to transform $s_i^{(in)}$ back into $c_i^{(in)}$ is at most $\delta_{in} m$, in which case $s_i^{(in)}$ gets decoded to the correct outer symbol using C_{in}.

However, we must also account for the fact that some runs r_j of $c_i^{(in)}$ may become deleted completely after duplication and passing through the deletion channel. That is, the image of r_j in s under $f^{(del)} \circ f^{(dup)}$ is empty, or, in other words, r_j is type-0. In this case the two neighboring runs r_{j-1} and r_{j+1} appear merged together in the Deduplication step of decoding. For example, if a run of 1s was deleted completely after duplication and deletion, its neighboring runs of 0s would be interpreted by the decoder as a single run. Fortunately, as we saw, the probability that a run is type-0 is extremely small ($< 10^{-10}$), and we show each type-0 run only increases $\Delta_{i/d}(c_i^{(in)}, s_i^{(in)})$ by a constant. We show this constant is at most 6.

To be precise, let Y_j be a random variable that is 0 if $|r_j| = t_j$, 1 if $\{|r_j|, t_j\} = \{1, 2\}$, and 6 if $t_j = 0$. We claim $\sum_{j=1}^{k} Y_j$ is an upper bound on $\Delta_{i/d}(c_i^{(in)}, s_i^{(in)})$. To see this, first note that if $t_j \neq 0$ for all i, then the number of runs of $c_i^{(in)}$ and $s_i^{(in)}$ are equal, so we can transform $c_i^{(in)}$ into $s_i^{(in)}$ by adding a bit to each length-1 type-2 run of $c_i^{(in)}$ and deleting a bit from each length-2 type-1 run of $s_i^{(in)}$.

Now, if some number, ℓ, of the t_j are 0, then at most 2ℓ of the runs in $c_i^{(in)}$ become merged with some other run (or a neighboring decoding buffer) after duplication and deletion. Each set of consecutive runs $r_j, r_{j+2}, \dots, r_{j+2j'}$ that are merged after duplication and deletion gets replaced with 1 or 2 copies of the corresponding bit. For example, if $r_1 = \langle 11 \rangle, r_2 = \langle 0 \rangle, r_3 = \langle 11 \rangle$, and if after duplication and deletion, $2B$ bits remain in the image of each of r_1 and r_3, and r_2 is type-0, then the image of $r_1 r_2 r_3$ under $f^{(del)} \circ f^{(dup)}$ is $\langle 1 \rangle^{4B}$, which gets decoded as $\langle 11 \rangle$ in the Deduplication step because $\langle 1 \rangle^{4B}$ is type-2. To account for the type-0 runs in transforming $c_i^{(in)}$ into $s_i^{(in)}$, we (i) delete at most two bits from each of the ℓ type-0 runs in $c_i^{(in)}$ and (ii) delete at most two bits for each of at most 2ℓ merged runs in $c_i^{(in)}$. The total number of additional insertions and deletions required to account for type-0 runs of c is thus at most 6ℓ, so we need at most 6 insertions and deletions to account for each type-0 run.

Our analysis covers the case when some bits in the image of $c_i^{(in)}$ under $f^{(del)} \circ f^{(dup)}$ are interpreted as part of a decoding buffer. Recall that inner codewords start and end with a 1, so that $r_1 \in \{\langle 1 \rangle, \langle 11 \rangle\}$ for every inner codeword. If, for example, $t_1 = 0$, that is, the image under $f^{(del)} \circ f^{(dup)}$ of the first run of 1s, r_1, is the empty string, then the bits of r_2 are interpreted as part of the decoding buffer. In this case too, our analysis tells us that the type-0 run r_1 increases $\Delta_{i/d}(c_i^{(in)}, s_i^{(in)})$ by at most 6.

We conclude $\sum_{j=1}^{k} Y_j$ is an upper bound for $\Delta_{i/d}(c_i^{(in)}, s_i^{(in)})$.

Note that if r_j has length 1, then by (5) we have

$$\mathbb{E}[Y_j] \;=\; 1 \cdot \mathbf{Pr}[r_j \text{ is type-2}] + 6 \cdot \mathbf{Pr}[r_j \text{ is type-0}] \;<\; 1 \cdot 0.0071 + 6 \cdot 10^{-9} \;<\; 0.0082. \quad (7)$$

Similarly, if r_j has length 2, then by (6) we have

$$\mathbb{E}[Y_j] \;=\; 1 \cdot \mathbf{Pr}[r_j \text{ is type-1}] + 6 \cdot \mathbf{Pr}[r_j \text{ is type-0}] \;<\; 1 \cdot 0.0081 + 6 \cdot 10^{-9} \;<\; 0.0082. \quad (8)$$

Thus $\mathbb{E}[Y_j] < 0.0082$ for all i. We know the word $s_i^{(in)}$ is decoded incorrectly (i.e. is not decoded as σ_i) in the Inner Decoding step only if $\Delta_{i/d}(c_i^{(in)}, s_i^{(in)}) > \delta_{in}m$. The Y_j are independent, so Lemma 3 gives

$$\begin{aligned}\mathbf{Pr}[s_i^{(in)} \text{ decoded incorrectly}] &\leq \mathbf{Pr}[Y_1 + Y_2 + \cdots + Y_k \geq \delta_{in}m] \\ &\leq \mathbf{Pr}[Y_1 + Y_2 + \cdots + Y_k \geq \delta_{in}k] \\ &\leq \exp\left(-\frac{(\delta_{in} - 0.0082)^2 k}{3 \cdot 6 \cdot \delta_{in}}\right) \\ &\leq \exp\left(-\Omega(m)\right) \qquad (9)\end{aligned}$$

where the last inequality is given by $k \geq m/2$. Since our $m \geq \Omega(\log(1/\delta_{out}))$ is sufficiently large, we have the probability $s_i^{(in)}$ is decoded incorrectly is at most $\delta_{out}/10$. If we let $Y_j^{(i)}$ denote the Y_j corresponding to inner codeword $c_i^{(in)}$, the events E_i given by $\sum_j Y_j^{(i)} \geq \delta_{in}m$ are independent. By concentration bounds on the events E_i, we conclude the probability that there are at least $\delta_{out}n/9$ incorrectly decoded inner codewords that are not already affected by spurious buffers and neighboring deleted buffers is $2^{-\Omega(n)}$.

Our aim is to show that the number of spurious buffers, deleted buffers, and inner decoding failures is small with high probability. So far, we have shown that, with high probability, assuming a codeword is not already affected by spurious buffers and neighboring deleted buffers, the number of inner decoding failures is small. We now turn to showing the number of spurious buffers is likely to be small.

A spurious buffer appears inside an inner codeword if many consecutive runs of 1s are type-0. A spurious buffer requires at least one of the following: (i) a codeword contains a sequence of at least $\eta m/5$ consecutive type-0 runs of 1s, (ii) a codeword contains a sequence of $\ell \leq \eta m/5$ consecutive type-0 runs of 1s, such that, for the $\ell + 1$ consecutive runs of 0s neighboring these type-0 runs of 1s, their image under $f^{(del)} \circ f^{(dup)}$ has at least $0.5\eta m$ 0s. We show both happen with low probability within a codeword.

A set of ℓ consecutive type-0 runs of 1s occurs with probability at most $10^{-10\ell}$. Thus the probability an inner codeword has a sequence of $\eta m/5$ consecutive type-0 runs of 1s is at most $m^2 \cdot 10^{-10\eta m/5} = \exp(-\Omega(\eta m))$. Now assume that in an inner codeword, each set of consecutive type-0 runs of 1s has size at most $\eta m/5$. Each set of ℓ consecutive type-0 runs of 1s merges $\ell + 1$ consecutive runs of 0s in c, so that they appear as a single longer run in s. The sum of the lengths of these $\ell + 1$ runs is some number ℓ^* that is at most $2\ell + 2$. The number of bits in the image of these runs of $c_i^{(in)}$ under $f^{(del)} \circ f^{(dup)}$ is distributed as $\text{Binomial}(\ell^* B/(1-p), 1-p)$. This has expectation $\ell^* B \leq 0.41 B\eta m$, so by concentration bounds, the probability this run of s has length at least $0.5B\eta m$, i.e. is interpreted as a decoding buffer, is at most $\exp(-\Omega(\eta m))$. Hence, conditioned on each set of consecutive type-0 runs of 1s having size at most $\eta m/5$, the probability of having no spurious buffers in a codeword is at least $1 - \exp(-\Omega(\eta m))$. Thus the overall probability there are no spurious buffers a given inner codeword is at least $(1 - \exp(-\Omega(\eta m))(1 - \exp(-\Omega(\eta m))) = 1 - \exp(-\Omega(\eta m))$. Since each inner codeword contains at most m candidate spurious buffers (one for each type-0 run of 1s), the expected number of spurious buffers in an inner codeword is thus at most $m \cdot \exp(-\Omega(\eta m))$. By our choice of $m \geq \Omega(\log(1/\delta_{out})/\eta)$, this is at most $\delta_{out}/10$. The occurrence of conditions (i) and (ii) above are independent between buffers. The total number of spurious buffers thus is bounded by the sum of n independent random variables each with expectation at most $\delta_{out}/10$. By concentration bounds, the probability that there are at least $\delta_{out}n/9$ spurious buffers is $2^{-\Omega(n)}$.

A deleted buffer occurs only when the image of the ηm 0s in a buffer under $f^{(del)} \circ f^{(dup)}$ is at most $B\eta m/2$. The number of such bits is distributed as Binomial$(B\eta m/(1-p), 1-p)$. Thus, each buffer is deleted with probability $\exp(-B\eta m) < \delta_{out}/10$ by our choice of $m \geq \Omega(\log(1/\delta_{out})/\eta)$. The events of a buffer receiving too many deletions are independent across buffers. By concentration bounds, the probability that there are at least $\delta_{out}n/9$ deleted buffers is thus $2^{-\Omega(n)}$.

Each inner decoding failure, spurious buffer, and deleted buffer increases the distance $\Delta_{i/d}(c_i^{(out)}, s_i^{(out)})$ by at most 3: each inner decoding failure causes up to 1 insertion and 1 deletion; each spurious buffer causes up to 1 deletion and 2 insertions; and each deleted buffer causes up to 2 deletions and 1 insertion. Our message is decoded incorrect if $\Delta_{i/d}(c_i^{(out)}, s_i^{(out)}) > \delta_{out}n$. Thus, there is a decoding error in the outer code only if at least one of (i) the number of incorrectly decoded inner codewords without spurious buffers or neighboring deleted buffers, (ii) the number of spurious buffers, or (iii) the number of deleted buffers is at least $\delta_{out}n/9$. However, by the above arguments, each is greater than $\delta_{out}n/9$ with probability $2^{-\Omega(n)}$, so there is a decoding error with probability $2^{-\Omega(n)}$. This concludes the proof of Theorem 1.

3.2 Possible Alternative Constructions

As mentioned in the introduction, Drinea and Mitzenmacher [4, 5] proved that the capacity of the BDC$_p$ is at least $(1-p)/9$. However, their proof is nonconstructive and they do not provide an efficient decoding algorithm.

One might think it is possible to use Drinea and Mitzenmacher's construction as a black box. We could follow the approach in this paper, concatenating an outer code given by [10] with the rate $(1-p)/9$ random-deletion-correcting code as a black box inner code. The complexity of the Drinea and Mitzenmacher's so-called *jigsaw decoding* is not apparent from [5]. However, the inner code has constant length, so construction, encoding, and decoding would be constant time. Thus, the efficiency of the inner code would not affect the asymptotic runtime.

The main issue with this approach is that, while the inner code can tolerate random deletions with probability p, inner codeword bits are *not* deleted in the concatenated construction according to a BDC$_p$; the 0 bits closer to the buffers between the inner codewords are deleted with higher probability because they might be "merged" with a buffer. For example, if an inner codeword is $\langle 101111\rangle$, then because the codeword is surrounded by buffers of 0s, deleting the leftmost 1 effectively deletes two bits because the 0 is interpreted as part of the buffer. While this may not be a significant issue because the distributions of deletions in this deletion process and BDC$_p$ are quite similar, much more care would be needed to prove correctness.

Our construction does not run into this issue, because our transmitted codewords tend to have *many* 1s on the ends of the inner codewords. In particular, each inner codeword of C_{in} has 1s on the ends, so after the Duplication step each inner codeword has $B/(1-p)$ or $2B/(1-p)$ 1s on the ends. The 1s on the boundary of the inner codeword will all be deleted with probability $\approx \exp(-B)$, which is small. Thus, in our construction, it is far more unlikely that bits are merged with the neighboring decoding buffer, than if we were to use a general inner code construction. Furthermore, we believe our construction based on bit duplication of a worst-case deletion correcting code is conceptually simpler than appealing to an existential code.

As a remark, we presented a construction with rate $(1-p)/110$, but using a randomized encoding we can improve the constant from $1/110$ to $1/60$. We can modify our construction so that, during the Duplication step of decoding, instead of replacing each bit of $c^{(cat)}$ with a fix number $B/(1-p)$ copies of itself, we instead replaced each bit independently with Poisson$(B/(1-p))$ copies of itself. Then the image of a run r_j under duplication and deletion is distributed as Poisson(B), which is independent of p. Because we don't have a dependence on p, we can tighten our bounding in (5) and (6). To obtain $(1-p)/60$, we can take $B = 28.12$ and set $B^* = 40$, where B^* is the threshold after which runs are decoded as two bits instead of one bit in the Deduplication step. The disadvantage of this approach is that we require our encoding to be randomized, whereas the construction presented above uses deterministic encoding.

4 Future work and open questions

A lemma due to Levenshtein [15] states that a code C can decode against pn adversarial deletions if and only if it can decode against pn adversarial insertions and deletions. While this does not automatically preserve the efficiency of the decoding algorithms, all the recent efficient constructions of codes for worst-case deletions also extend to efficient constructions with similar parameters for recovering from insertions and deletions [1, 7].

In the random error model, decoding deletions, insertions, and insertions and deletions are not the same. Indeed, it is not even clear how to define random insertions. One could define insertions and deletions via the Poisson repeat channel where each bit is replaced with a Poisson many copies of itself (see [4, 17]). However, random insertions do not seem to share the similarity to random deletions that adversarial deletions share with adversarial insertions; we can decode against arbitrarily large Poisson duplication rates, whereas for codes of block length n we can decode against a maximum of n adversarial insertions or deletions [5]. Alternatively one can consider a model of random insertions and deletions where, for every bit, the bit is deleted with a fixed probability p_1, a bit is inserted after it with a fixed probability p_2, or it is transmitted unmodified with probability $1-p_1-p_2$ [19]. One could also investigate settings involving memoryless insertions, deletions, and substitutions [16].

There remain a number of open questions even concerning codes for deletions only. Here are a few highlighted by this work.

1. Can we close the gap between $\sqrt{2}-1$ and $\frac{1}{2}$ on the maximum correctable fraction of adversarial deletions?
2. Can we construct efficiently decodable codes for the binary deletion channel with better rate, perhaps reaching or beating the best known existential capacity lower bound of $(1-p)/9$?
3. Can we construct efficient codes for the binary deletion channel with rate $1-O(h(p))$ for $p \to 0$?

References

1 Joshua Brakensiek, Venkatesan Guruswami, and Samuel Zbarsky. Efficient low-redundancy codes for correcting multiple deletions. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1884–1892, 2016.

2 Boris Bukh, Venkatesan Guruswami, and Johan Håstad. An improved bound on the fraction of correctable deletions. *IEEE Trans. Information Theory*, 63(1):93–103, 2017.

3 Suhas Diggavi and Matthias Grossglauser. On transmission over deletion channels. In *Proceedings of the 39th Annual Allerton Conference on Communication, Control, and Computing*, pages 573–582, 2001.

4 Eleni Drinea and Michael Mitzenmacher. On lower bounds for the capacity of deletion channels. *IEEE Transactions on Information Theory*, 52(10):4648–4657, 2006.

5 Eleni Drinea and Michael Mitzenmacher. Improved lower bounds for the capacity of i.i.d. deletion and duplication channels. *IEEE Trans. Information Theory*, 53(8):2693–2714, 2007.

6 Dario Fertonani, Tolga M. Duman, and M. Fatih Erden. Bounds on the capacity of channels with insertions, deletions and substitutions. *IEEE Trans. Communications*, 59(1):2–6, 2011. doi:10.1109/TCOMM.2010.110310.090039.

7 Venkatesan Guruswami and Ray Li. Efficiently decodable insertion/deletion codes for high-noise and high-rate regimes. In *IEEE International Symposium on Information Theory, ISIT 2016, Barcelona, Spain, July 10-15, 2016*, pages 620–624, 2016. doi:10.1109/ISIT.2016.7541373.

8 Venkatesan Guruswami and Ray Li. Coding against deletions in oblivious and online models, 2017. Manuscript; arXiv abs/1612.06335. URL: http://arxiv.org/abs/1612.06335.

9 Venkatesan Guruswami and Carol Wang. Deletion codes in the high-noise and high-rate regimes. *IEEE Trans. Information Theory*, 63(4):1961–1970, 2017. doi:10.1109/TIT.2017.2659765.

10 Bernhard Haeupler and Amirbehshad Shahrasbi. Synchronization strings i: Codes for insertions and deletions approaching the singleton bound. *To appear in STOC'17.* http://arxiv.org/abs/1704.00807.

11 Adam Kalai, Michael Mitzenmacher, and Madhu Sudan. Tight asymptotic bounds for the deletion channel with small deletion probabilities. In *IEEE International Symposium on Information Theory, ISIT 2010, June 13-18, 2010, Austin, Texas, USA, Proceedings*, pages 997–1001, 2010. doi:10.1109/ISIT.2010.5513746.

12 Yashodhan Kanoria and Andrea Montanari. Optimal coding for the binary deletion channel with small deletion probability. *IEEE Trans. Information Theory*, 59(10):6192–6219, 2013. doi:10.1109/TIT.2013.2262020.

13 Ian Kash, Michael Mitzenmacher, Justin Thaler, and John Ullman. On the zero-error capacity threshold for deletion channels. In *Information Theory and Applications Workshop (ITA)*, pages 1–5, January 2011.

14 Adam Kirsch and Eleni Drinea. Directly lower bounding the information capacity for channels with i.i.d.deletions and duplications. *IEEE Trans. Information Theory*, 56(1):86–102, 2010. doi:10.1109/TIT.2009.2034883.

15 Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Dokl. Akad. Nauk*, 163(4):845–848, 1965 (Russian). English translation in Soviet Physics Doklady, 10(8):707-710, 1966.

16 Hugues Mercier, Vahid Tarokh, and Fabrice Labeau. Bounds on the capacity of discrete memoryless channels corrupted by synchronization and substitution errors. *IEEE Trans. Information Theory*, 58(7):4306–4330, 2012. doi:10.1109/TIT.2012.2191682.

17 Michael Mitzenmacher. A survey of results for deletion channels and related synchronization channels. *Probability Surveys*, 6:1–33, 2009.

18 Mojtaba Rahmati and Tolga M. Duman. Upper bounds on the capacity of deletion channels using channel fragmentation. *IEEE Trans. Information Theory*, 61(1):146–156, 2015. doi:10.1109/TIT.2014.2368553.

19 Ramji Venkataramanan, Sekhar Tatikonda, and Kannan Ramchandran. Achievable rates for channels with deletions and insertions. *IEEE Trans. Information Theory*, 59(11):6990–7013, 2013. doi:10.1109/TIT.2013.2278181.

20 S. M. Sadegh Tabatabaei Yazdi and Lara Dolecek. A deterministic polynomial-time protocol for synchronizing from deletions. *IEEE Trans. Information Theory*, 60(1):397–409, 2014. `doi:10.1109/TIT.2013.2279674`.

A Proof of Lemma 3

Proof. For each i, we can find a random variable B_i such that $B_i \geq A_i$ always, B_i takes values in $[0, \beta]$, and $\mathbb{E}[B_i] = \alpha$. Applying Lemma 2 gives

$$\begin{aligned}\mathbf{Pr}\left[\sum_{i=1}^{n} A_i \geq n\gamma\right] &\leq \mathbf{Pr}\left[\sum_{i=1}^{n} B_i \geq n\gamma\right]\\ &\leq \mathbf{Pr}\left[\sum_{i=1}^{n} \frac{B_i}{\beta} \geq \left(1 + \left(\frac{\gamma-\alpha}{\alpha}\right)\right)\frac{n\alpha}{\beta}\right]\\ &\leq \exp\left(-\frac{\left(\frac{\gamma-\alpha}{\alpha}\right)^2 \cdot \frac{n\alpha}{\beta}}{3}\right)\\ &= \exp\left(-\frac{(\gamma-\alpha)^2 n}{3\alpha\beta}\right).\end{aligned}$$

◀

On Some Computations on Sparse Polynomials

Ilya Volkovich

Department of EECS, University of Michigan, Ann Arbor, MI, USA
ilyavol@umich.edu

Abstract

In arithmetic circuit complexity the standard operations are $\{+, \times\}$. Yet, in some scenarios exponentiation gates are considered as well (see e.g. [6, 1, 28, 30]). In this paper we study the question of efficiently evaluating a polynomial given an oracle access to its power. Among applications, we show that:

- A reconstruction algorithm for a circuit class $\mathcal{C}$ can be extended to handle f^e for $f \in \mathcal{C}$.
- There exists an efficient deterministic algorithm for factoring sparse multiquadratic[1] polynomials.
- There is a deterministic algorithm for testing a factorization of sparse polynomials, with constant individual degrees, into sparse irreducible factors. That is, testing if $f = g_1 \cdot \ldots \cdot g_m$ when f has constant individual degrees and g_i-s are irreducible.
- There is a deterministic reconstruction algorithm for multilinear[2] depth-4 circuits with two multiplication gates.
- There exists an efficient deterministic algorithm for testing whether two powers of sparse polynomials are equal. That is, $f^d \equiv g^e$ when f and g are sparse.

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

Keywords and phrases Derandomization, Arithmetic Circuits, Reconstruction

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.48

1 Introduction

Let $f(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be a polynomial over the field $\mathbb{F}$. In this paper we study the following question: given $e \in \mathbb{N}$ and an oracle access to $f^e \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ can we efficiently implement an oracle access to f? That is, we wish to evaluate f on a set of points $\bar{a}, \bar{b}, \ldots$ (which might be unknown upfront) given an oracle access to f^e. An efficient *randomized* algorithm for this problem was given in [23]. Where, in fact, a randomized polynomial factorization algorithm was given. In addition, in terms of circuit complexity, it was shown in [43, 20] that if f^e has a small circuit then so does f, when the characteristic of $\mathbb{F}$ is zero or coprime with e.

For our applications, we only need to solve the problem in the oracle model, yet *deterministically*. Although, it is conceivable that the techniques of [43, 20] could work in oracle model, they will still be subject to the co-primality condition. In this paper we solve the problem for any e.

It is clear that as the first step, we should be able to extract e-th roots of field elements. For instance, if f is constant. We refer to such an algorithm as an e-th *root oracle* R_e. However, having root oracles is not enough for our task as demonstrated by the following example.

[1] A polynomial is multiquadratic if the degree of each variable is at most 2.
[2] A polynomial is multilinear if the degree of each variable is at most 1.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 48; pp. 48:1–48:21
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Let $h(x) = 3x - 4$ and $f = h^2$. Suppose that we wish to evaluate $h(x)$ at $x = 1, 2$ given an oracle access to $f(x)$ and using a square-root oracle R_2. As $f(1) = 1, f(2) = 4$ the oracle might return $h(1) = R_2(1) = 1$ and $h(2) = R_2(4) = 2$ (for example, returning the positive root). Note, however, that these evaluations are inconsistent with either $\pm h$! More generally, there could be e different $h_1, \ldots h_e$ polynomials resulting in the same polynomial when raised the e-th power (i.e. $\forall i \in [n] : h_i^e = f$). Therefore, in order to prevent the aforementioned situation our algorithm should output an oracle access to exactly one of them. We prove the following theorem.

▶ **Theorem 1** (Technical Contribution)**.** *There exists a deterministic algorithm that given $e \in \mathbb{N}$, an e-th root oracle R_e and an oracle access to a polynomial $f^e \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ of degree at most d uses* $\text{poly}(n, d, e, \log|\mathbb{F}|)$ *field operations and oracle calls to R_e, and outputs an oracle access to $\omega \cdot f$, where $\omega \in \mathbb{F}$ is such that $\omega^e = 1$.*

We note that similar ideas appeared previously in the literature, although partially and implicitly. The problem can seen as a version of list-decoding of Reed-Muller codes. Indeed, mirroring the list-decoding algorithm of [42] and the factorization algorithm of [23], the proposed algorithm uses an anchor point and draws a line to that point in order to choose the correct answer from a small list of possible answers. We now discuss related problems and applications.

1.1 Multivariate Polynomial Factorization

One of the fundamental problems in algebraic complexity is the problem of polynomial factorization: given a polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ over a field $\mathbb{F}$, find its irreducible factors. Other than being natural, the problem has many applications such as list decoding [41, 17] and derandomization [19]. A large amount of research has been devoted to finding efficient algorithms for this problem (see e.g. [48]) and numerous *randomized* algorithms were designed [49, 20, 21, 23, 48, 22, 47]. However, the question of whether there exist *deterministic* algorithms for this problem remains an interesting open question (see [48, 27]).

Perhaps the simplest factorization algorithm is a root oracle. We note that the best known *deterministic* root extraction algorithms over the finite fields have polynomial dependence on the field characteristic p (see e.g. [36, 48, 14, 27]). While in the *randomized* setting, this dependence is polynomial in $\log p$. In particular, there is no known efficient deterministic root extraction algorithm when p is large. Over fields with characteristic 0 (e.g. $\mathbb{Q}$) both the *deterministic* and the *randomized* complexities are polynomial in the bit-complexity of the coefficients (see [31]). Therefore, we can say that root extraction is, perhaps, the simplest hard problem in polynomial factorization. For sake of uniformity we formulate all our results in terms of root oracles and $\log|\mathbb{F}|$ which stands for the bit-complexity of the coefficients in the underlying polynomials.

1.2 Polynomial Reconstruction

Let $\mathbb{F}$ be a field and $\mathcal{C}$ a class of circuits. The *reconstruction* problem for the class $\mathcal{C}$ is defined as follows. Given an oracle access to a polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$, computable by a circuit from $\mathcal{C}$, output a circuit $C \in \mathcal{C}$ that computes f. A reconstruction algorithm is *efficient* if the number of queries it makes to f and its running time are polynomial in the size of the representation of f in the class $\mathcal{C}$. The reconstruction problem can be seen as the algebraic analog of the learning problem.

An immediate application of our main theorem is reconstruction beyond an exponentiation gate. More formally, we can efficiently extend a reconstruction algorithm for a circuit class $\mathcal{C}$ to handle polynomials of the form f^e when f is computable by a circuit $C \in \mathcal{C}$. Note that in general f^e might not be computable by a circuit in $\mathcal{C}$.

▶ **Theorem 2.** *Let A be a deterministic (randomized) reconstruction algorithm for a circuit class $\mathcal{C}$, let $f \in \mathcal{C}$ and let $T(f)$ denote the number of operations A uses to reconstruct f. Then there exists a deterministic (randomized) algorithm that given $e \in \mathbb{N}$, an e-th root oracle R_e and an oracle access to the polynomial $f^e \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ of degree at most d, uses* $\mathrm{poly}(n, d, \log|\mathbb{F}|, T(f))$ *field operations and oracles calls to R_e and A, and outputs a circuit for $\omega \cdot f$, where $\omega \in \mathbb{F}$ is such that $\omega^e = 1$.*

As a corollary we get to extend reconstruction algorithms for specific classes of circuits. An *s-sparse polynomial* is polynomial with at most s (non-zero) monomials. Sparse polynomials were deeply studied (see e.g. [5, 29, 32]) and, in fact, several efficient deterministic reconstruction algorithms were given. Our next result extends the reconstruction algorithm of [29] to powers of sparse polynomials.

▶ **Theorem 3.** *Let $n, s, d, e \in \mathbb{N}$ and let $f(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be an s-sparse polynomial of degree at most d. Then there exists a deterministic algorithm that given $e \in \mathbb{N}$, an oracle access to the polynomial $f^e \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and an e-th root oracle R_e uses* $\mathrm{poly}(n, d, e, s, \log|\mathbb{F}|)$ *field operations and oracles calls, and outputs $\omega \cdot f$, where $\omega \in \mathbb{F}$ is such that $\omega^e = 1$.*

Read-once formulas are formulas in which each variable appears at most once. A *read-once polynomial* is a polynomial computable by a read-once formula. Those are the smallest possible polynomials that depend on all of their variables. Although they form a very restricted model of computation, read-once formulas received a lot of attention [18, 25, 3, 8, 6, 7, 38, 39, 33, 45]. In [38] a $n^{\mathcal{O}(\log n)}$-time reconstruction algorithm for read-once formulas was given. In [33], the runtime of the algorithm was improved to $\mathrm{poly}(n)$. Our next result extends the reconstruction algorithm further to powers of read-once polynomials. We note that the reconstruction algorithm of [6] actually deals with a richer model of read-once formulas with exponentiation gates. Yet, that algorithm is randomized.

▶ **Theorem 4.** *Let $n, e \in \mathbb{N}$ and let $f(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be a read-once polynomial. Then there exists a deterministic algorithm that given an oracle access to the polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and an e-th root oracle R_e uses* $\mathrm{poly}(n) \cdot \mathrm{poly}(e, \log|\mathbb{F}|)$ *field operations and oracles calls, and outputs a read-once formula Ψ that computes $\omega \cdot f$, where $\omega \in \mathbb{F}$ is such that $\omega^e = 1$.*

A *depth*-4 $\Sigma\Pi\Sigma\Pi(k)$ circuit has 4 layers of alternating $(+, \times)$ gates and it computes a polynomial of the form $C(x_1, x_2, \cdots, x_n) = \sum_{i=1}^{k} F_i = \sum_{i=1}^{k} \prod_{j=1}^{d_i} P_{ij}$ where k is the fan-in of the top Σ gate and d_i are the fan-ins of the Π gates at the second level. These circuits were previously studied in [2, 16, 26, 35]. In particular, in [16] a randomized reconstruction algorithm was given for multilinear depth-4 circuits with $k = 2$ (i.e. $\Sigma\Pi\Sigma\Pi(2)$ circuits). As an application, we derandomize their algorithm using a square root oracle. We note that our result achieves an optimal derandomization since in [46] it was shown that any reconstruction algorithm for this circuit class must compute square roots.

▶ **Theorem 5.** *Let $n, s \in \mathbb{N}$ and suppose* $\mathrm{char}(\mathbb{F}) \neq 2$. *Then there exists a deterministic algorithm that given an oracle access to the polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ computable by a multilinear* $\Sigma\Pi\Sigma\Pi(2)$ *circuit of size s and a square root oracle R_2 uses* $\mathrm{poly}(n, s, \log|\mathbb{F}|)$ *field operations and oracles calls, and outputs a* $\Sigma\Pi\Sigma\Pi(2)$ *circuit that computes f.*

1.3 Sparse Polynomial Factorization

Coming up with an efficient deterministic factorization algorithm for sparse polynomials (given as a list of monomials) is a classical open question posed by von zur Gathen and Kaltofen in [49]. An inherent difficulty in tackling the problem lies within the fact that a factor of a sparse polynomial need not be sparse. Example 5.1 in [49] demonstrates that a blow-up in the sparsity of a factor can be super-polynomial over any field. Consequently, just writing down the irreducible factors as lists of monomials can take super-polynomial time. In fact, the randomized algorithm of [49] assumes that the upper bound on the sparsity of the factors is known. In light of this difficulty, a simpler problem was posed in that same paper: Given $m+1$ sparse polynomials $f, g_1, g_2, \dots g_m$ test if $f = g_1 \cdot g_2 \cdot \ldots \cdot g_m$. This problem is referred to as "testing sparse factorization".

Our main result gives a deterministic factorization algorithm for sparse multiquadratic polynomials.

▶ **Theorem 6.** *Let $n, s \in \mathbb{N}$ and suppose $\mathrm{char}(\mathbb{F}) \neq 2$. There exists a deterministic algorithm that given an s-sparse multiquadratic polynomial $f(\bar{x}) \in \mathbb{F}[x_1, x_2, \dots, x_n]$ and a square root oracle R_2 uses $\mathrm{poly}(n, s, \log|\mathbb{F}|)$ field operations and oracle calls to R_2 and outputs the irreducible factors of $f(\bar{x})$. That is, a list $h_1, \dots, h_k$ of irreducible polynomials such that $f = h_1 \cdot \ldots \cdot h_k$.*

We also show how to test sparse factorization for a special case of polynomials with constant individual degrees.

▶ **Theorem 7.** *Let $f, g_1, \dots g_m \in \mathbb{F}[x_1, x_2, \dots, x_n]$ be s-sparse polynomials a let d be a bound on the individual degrees of f. Then given $f, g_1, \dots g_m$, there exists a deterministic algorithm that tests if $f = g_1 \cdot g_2 \cdot \ldots \cdot g_m$ using $\mathrm{poly}(n, s^d, \log|\mathbb{F}|)$ field operations.*

Using techniques from Differential Field Theory we show that some identity testing algorithms could be extended to work beyond an exponentiation gate. In particular, we prove the following theorem which can be seen as testing symmetric sparse factorization. We note that setting $e = 1$ instantiates to testing sparse factorization in the case when $f_1 = f_2 = \ldots = f_m$.

▶ **Theorem 8.** *Let $n, s, d, e, \delta \in \mathbb{N}$ and let $f(\bar{x}), g(\bar{x}) \in \mathbb{F}[x_1, x_2, \dots, x_n]$ be two s-sparse polynomials of degree at most δ. Furthermore, suppose that $\mathrm{char}(\mathbb{F}) = 0$ or $\mathrm{char}(\mathbb{F}) > \delta \cdot \min(e, d)$. Then there exists a deterministic algorithm that given f, g, d and e uses $\mathrm{poly}(n, s, d, e, \delta, \log|\mathbb{F}|)$ field operations and tests whether $f^d = g^e$.*

We note that similar results to Theorems 7 and 8 follow from the works of [1, 4]. For the result of Theorem 8 we give a more direct and simple algorithm.

1.4 Techniques

Our main technique is to convert an oracle access to a power of a polynomial f^e into an oracle access to the polynomial itself f. As was discussed in the first part of the Introduction, a necessarily condition is having an efficient root extraction algorithm for field elements, referred to as a "root oracle". Yet, as was demonstrated further, applying root oracles naivly can result in inconsistency. More specifically, as there could be e roots of a polynomial, differing only by a multiplicative factor of a root of unity of order e, a root oracle can mismatch the answers to different oracle queries. We solve this problem by introducing an anchor and matching all the queries to that anchor. More specifically, we fix a non-zero assignment $\bar{a}$ of f. For query point $\bar{b}$ we compute the root along the line $\ell_{\bar{a},\bar{b}}(t)$ that passes through $\bar{a}$ and $\bar{b}$. Thus, we reduce the problem from n variables to 1. Finally, we show how to use a root oracle to compute a root of a univariate polynomial. The latter is carried out via Squarefree decomposition. See Sections 2.5 and 4.1 for more details.

In order to deal with sparse multiquadratic polynomials, we first show that a factor of such a polynomial is also sparse. Next, we apply the quadratic formula to get explicit expressions for the factors. Yet, these expression involve square roots. Computing a square root of a polynomial h can be seen as computing $\pm f$ given $h = f^2$. To this end, we first apply our main technique to get an oracle access for f and then use a reconstruction algorithm for sparse polynomials to compute the polynomial. See Section 4.3 for more details.

Another tool that we use is Resultants and Subresultants. These objects have seen various applications in algebraic complexity, computer algebra, elimination theory and other areas (see e.g. [15, 48, 10]). In particular, these are used to test coprimality of polynomials. We show how to efficiently employ them with sparse polynomial of constant degree. The main observation is that a resultant of two sparse polynomials of constant degrees is also a somewhat sparse polynomial of a "small" degree. For more details see Sections A and 2.4.

1.5 Previous Results

Over the last three decades the question of derandomizing sparse polynomial factorization has seen only a very partial progress. In [37], Shpilka & Volkovich gave efficient deterministic factorization algorithms for sparse multilinear polynomials. This result was extended in [44] to the model of sparse polynomials that split into multilinear factors. For the testing version of the problem, Saha et al. [34] presented an efficient deterministic algorithm for the special case when the sparse polynomials are sums of univariate polynomials.

1.6 Organization

We begin by some basic definitions and notation in Section 2 when in Section 2.5 we show how to compute a root of a univariate polynomial. In Section 3 we discuss sparse polynomials, their properties and some related efficient algorithms which leverage these properties. In particular, in Section 3.1 we prove that a factor of a sparse multiquadratic polynomial is also sparse. In Section 4 we give all our results showing how to perform certain computations on polynomials given an oracle access to their powers. We begin (Section 4.1) by showing how convert an oracle access to f^e into an oracle access to f using an e-th root oracle, thus proving Theorem (Theorem 1) which is our main technical contribution. The first application is given in Section 4.2 where we show how to extend a reconstruction algorithm for a circuit class $\mathcal{C}$ to handle powers of polynomials from $\mathcal{C}$ (Theorem 2). As a corollary, we obtain an efficient reconstruction algorithm for powers of sparse (Theorem 3) and read-once (Theorem 4) polynomials. Our main application is given in Section 4.3 where we present the first efficient factorization algorithm for sparse multiquadratic polynomials, thus proving theorem Theorem 6. In Section C, using different techniques but following the general line, we show how certain polynomial identity testing algorithms can be extended to handle powers of polynomials. We conclude the paper with discussion and open questions in Section 5.

2 Preliminaries

Let $\mathbb{F}$ denote a field, finite or otherwise, and let $\overline{\mathbb{F}}$ denote its algebraic closure.

2.1 Polynomials

A polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ *depends* on a variable x_i if there are two inputs $\bar{\alpha}, \bar{\beta} \in \overline{\mathbb{F}}$ differing only in the i^{th} coordinate for which $f(\bar{\alpha}) \neq f(\bar{\beta})$. We denote by $\mathrm{var}(f)$ the set of variables that f depends on. We say that f is g are *similar* and denote by it $f \sim g$ if $f = \alpha g$ for some $\alpha \neq 0 \in \mathbb{F}$. For a polynomial $f(x_1, \ldots, x_n)$, a variable x_i and a field element α, we denote with $f|_{x_i=\alpha}$ the polynomial resulting from substituting α to x_i. Similarly given a subset $I \subseteq [n]$ and an assignment $\bar{a} \in \mathbb{F}^n$, we define $f|_{\bar{x}_I=\bar{a}_I}$ to be the polynomial resulting from substituting a_i to x_i for every $i \in I$.

▶ **Definition 9** (Line). Given $\bar{a}, \bar{b} \in \mathbb{F}^n$ we define a *line* passing through $\bar{a}$ and $\bar{b}$ as $\ell_{\bar{a},\bar{b}} : \mathbb{F} \to \mathbb{F}^n$, $\ell_{\bar{a},\bar{b}}(t) \triangleq (1-t) \cdot \bar{a} + t \cdot \bar{b}$. In particular, $\ell_{\bar{a},\bar{b}}(0) = \bar{a}$ and $\ell_{\bar{a},\bar{b}}(1) = \bar{b}$.

▶ **Definition 10** (Degrees, Leading Monomials, Leading Coefficients). The *leading monomial* of a polynomial f, $\mathrm{lm}(f)$ is defined as the largest non-zero monomial of f (with its coefficient) with respect to the lexicographical order of the monomials. The *total degree* of f is the largest total degree of a monomial in f. Let $x_i \in \mathrm{var}(f)$. We can write: $f = \sum_{j=0}^{d} f_j \cdot x_i^j$ such that $\forall j, x_i \notin \mathrm{var}(f_j)$ and $f_d \not\equiv 0$. The *leading coefficient* of f w.r.t to x_i is defined as $\mathrm{lc}_{x_i}(f) \triangleq f_d$. The *individual degree* of x_i in f is defined as $\deg_{x_i}(f) \triangleq d$.

It easy to see that for every $f, g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and $i \in [n]$ we have that: $\mathrm{lm}(f \cdot g) = \mathrm{lm}(f) \cdot \mathrm{lm}(g)$ and $\mathrm{lc}_{x_i}(f \cdot g) = \mathrm{lc}_{x_i}(f) \cdot \mathrm{lc}_{x_i}(g)$.

2.2 Partial Derivatives

The concept of a *partial derivative* of a multivariate function and its properties are well-known and well-studied for continuous domains (such as, $\mathbb{R}$, $\mathbb{C}$ etc.). This concept can be extended to polynomials and rational functions over arbitrary fields from a purely algebraic point of view. For more details we refer to reader to [24].

▶ **Definition 11.** For a monomial $M = \alpha \cdot x_1^{e_1} \cdots x_i^{e_i} \cdots x_n^{e_n} \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and a variable x_i we define the *partial derivative* of M with respect to x_i, as $\frac{\partial M}{\partial x_i} \triangleq \alpha e_i \cdot x_1^{e_1} \cdots x_i^{e_i - 1} \cdots x_n^{e_n}$. The definition can be extended to $\mathbb{F}[x_1, x_2, \ldots, x_n]$ by imposing linearity and to $\mathbb{F}(x_1, x_2, \ldots, x_n)$ via the quotient rule.

Observe that the sum, product, quotient and chain rules carry over. In addition, when $\mathbb{F} = \mathbb{R}$ or $\mathbb{F} = \mathbb{C}$ the definition coincides with the analytical one. The following set of rational function plays an important role.

▶ **Definition 12** (Field of Constants). The *Field of Constants* of $\mathbb{F}(x_1, x_2, \ldots, x_n)$ is defined as $\mathrm{C}(\mathbb{F}(x_1, x_2, \ldots, x_n)) \triangleq \left\{ f \in \mathbb{F}(x_1, x_2, \ldots, x_n) \;\middle|\; \forall i \in [n], \frac{\partial f}{\partial x_i} \equiv 0 \right\}$.

It is easy to see that the field of constants is, indeed, a field and in particular $\mathbb{F} \subseteq \mathrm{C}(\mathbb{F}(x_1, x_2, \ldots, x_n))$. Furthermore, this containment is proper for fields with positive characteristics and equality holds only for fields with characteristic 0. The following Lemma gives a precise characterization of $\mathrm{C}(\mathbb{F}(x_1, x_2, \ldots, x_n))$.

▶ **Lemma 13.** *Let $\mathbb{F}$ be a field of characteristic p. Then for every $n \in \mathbb{N}$:*

1. $\mathrm{C}(\mathbb{F}(x_1, x_2, \ldots, x_n)) = \mathbb{F}$ *when* $p = 0$.
2. $\mathrm{C}(\mathbb{F}(x_1, x_2, \ldots, x_n)) = \mathbb{F}(x_1^p, x_2^p, \ldots, x_n^p)$ *when p is positive.*

2.3 Factors and Perfect Powers

Let $f, g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be polynomials. We say that g *divides* f, or equivalently g is a factor of f, and denote it by $g \mid f$ if there exists a polynomial $h \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ such that $f = g \cdot h$. We say that f is *irreducible* if f is non-constant and cannot be written as a product of two non-constant polynomials. For $e \in \mathbb{N}$, we say that f is a *perfect e-th power* if there exists a polynomial $h \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ such that $f = h^e$. Equivalently, we say that h is f's e-th root. Given the notion of divisibility we define the gcd of a set of polynomials in the natural way. Given the notion of irreducibility we can state the important property of the uniqueness of factorization,

▶ **Lemma 14** (Uniqueness of Factorization). *Let $h_1^{e_1} \cdot \ldots \cdot h_k^{e_k} = g_1^{e'_1} \cdot \ldots \cdot g_{k'}^{e'_{k'}}$ be two factorizations of the same non-zero polynomial into irreducible, pairwise comprise factors. Then $k = k'$ and there exists a permutation $\sigma : [k] \to [k]$ such that $h_i \sim g_{\sigma(i)}$ and $e_i = e'_{\sigma(i)}$ for $i \in [k]$.*

By definition, the ratio α/β of two e-th of roots a field element (i.e. $\alpha^e = \beta^e \neq 0$) is a root of unity of order e. We show that the same holds for perfect roots of polynomials. More precisely, two e-th roots of the same polynomial differ only by a multiplicative factor ω satisfying $\omega^e = 1$.

▶ **Lemma 15.** *Let $f(\bar{x}), h(\bar{x}), g(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be polynomials such that $f(\bar{x}) = h(\bar{x})^e = g(\bar{x})^e$ for some $e \in \mathbb{N}$. In addition, let $\alpha \in \mathbb{F}, \bar{a} \in \mathbb{F}^n$ such that $\alpha^e = f(\bar{a}) \neq 0$. Then*

1. *There exists $\omega \in \mathbb{F}$ such that $\omega^e = 1$ and $h(\bar{x}) = \omega \cdot g(\bar{x})$.*
2. *There exists a unique polynomial $u(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ s.t. $f(\bar{x}) = u(\bar{x})^e$ and $u(\bar{a}) = \alpha$.*

The proof can be found in Section D.

2.4 GCD and Subresultants

As was mentioned earlier, the notion of divisibility gives rise to the notion of a gcd of a set of polynomials in the natural way. Furthermore, the uniqueness of factorization property of the rings of polynomials $\mathbb{F}[x_1, x_2, \ldots, x_n]$ ensures that a gcd is defined up to a multiplication by a field element. We can also consider versions of gcd when we concentrate on a single variable and treat the remaining variables as field elements. That is, given $f_1, \ldots, f_m$ consider $\gcd_{x_i}(f_1, \ldots, f_m)$. Naturally, such gcd's is defined up to a multiplication by a rational function depending on the remaining variables. Yet, in all such gcd's the variable x_i has the same degree.

▶ **Example 16.** Let $f = x_1^2x_2^2 + x_1^2x_2 + x_1x_2^2 + x_1x_2$ and $g = x_1^2x_2^2$. $\gcd(f, g) = x_1x_2$ while $\gcd_{x_1}(f, g) = x_1$. Yet, $\deg_{x_i}(\gcd_{x_i}(f, g)) = \deg_{x_i}(\gcd(f, g)) = 1$.

▶ **Lemma 17.** *Let $f, g \not\equiv 0 \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and let e_i denote the individual degree of x_i in g. Then $g \mid f$ iff $\forall i$ with $e_i > 0 : \deg_{x_i}(\gcd_{x_i}(f, g)) = e_i$.*

Proof. If $g \mid f$ then the statement is clear. Suppose $g \nmid f$. Let $g = \prod g_j^{d_j}$ be a factorization of g into irreducible, pairwise comprise factors. By definition, there exists j such that $g_j^{d_j} \nmid f$. Let $x_i \in \text{var}(g_j)$. As such, $\deg_{x_i}(\gcd_{x_i}(f, g)) \leq e_i - \deg_{x_i}(g_j) < e_i$. ◀

▶ **Definition 18** (Subresultant - Definition 7.3 from [15]). Let $f, g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be polynomials. Fix $i \in [n]$ and let d and e denote the degree of the variable x_i in f and g, respectively. We can write: $f = \sum_{j=0}^{d} f_j \cdot x_i^j$ such that $\forall j, x_i \notin \text{var}(f_j)$ and $g = \sum_{k=0}^{e} g_k \cdot x_i^k$

such that $\forall k, x_i \notin \text{var}(g_k)$. For $0 \leq j \leq \min\{e, d\}$ the *j-th Subresultant of f and g w.r.t x_i*, $S_{x_i}(j, f, g)$ is defined as a determinant of the $(d+e-2j) \times (d+e-2j)$ minor of the Sylvester Matrix of f and g. That is, the entities of the matrix are f_j-s and g_k-s.

Below is the crucial property of subresultants:

▶ **Lemma 19** (Lemma 7.1 and Theorem 7.3 from [15]). *For every variable x_i, the degree of x_i in $\gcd_{x_i}(f, g)$ equals to smallest j such that $S_{x_i}(j, f, g) \not\equiv 0$. In addition, if $u, v \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ such that $x_i \notin \text{var}(u) \cup \text{var}(v)$ then $\forall i, j$: $S_{x_i}(j, uf, vg) = S_{x_i}(j, f, g) \cdot u^{\deg_{x_i}(g)} \cdot v^{\deg_{x_i}(f)}$.*

Combining Lemmas 17 and 19 gives the following:

▶ **Corollary 20.** *Let $f, g \not\equiv 0 \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and let e_i denote the individual degree of x_i in g. Then $g \mid f$ iff $\forall i$ with $e_i > 0 : S_{x_i}(e_i - 1, f, g) \equiv 0$.*

Proof. If $g \mid f$ and $e_i > 0$, then $\deg_{x_i}(\gcd_{x_i}(f, g)) = e_i$ and thus $S_{x_i}(e_i, f, g) \not\equiv 0$ while $S_{x_i}(e_i - 1, f, g) \equiv 0$. On the other hand, if $S_{x_i}(e_i - 1, f, g) \equiv 0$ it must be the case that $S_{x_i}(e_i, f, g) \not\equiv 0$ since $\deg_{x_i}(\gcd_{x_i}(f, g)) \leq e_i$. ◀

2.5 Univariate Polynomials: Squarefree Decomposition and Root Computation

In this section we show how to compute the e-th roots of univariate polynomials using root oracles. We begin by discussing a Squarefree Decomposition of a polynomial. This is one of the steps in the majority of the polynomial factorization algorithms.

▶ **Definition 21** (Squarefree polynomials). We say that a polynomial $f(y) \in \mathbb{F}[y]$ is *squarefree* if $g(y)^2 \not| f(y)$ for every $g(y) \in \mathbb{F}[y]$.

▶ **Definition 22** (Squarefree Decomposition). Let $f(y) \in \mathbb{F}[y]$ be polynomial of degree at most d. The *squarefree decomposition* of $f(y)$ is a sequence of pairwise coprime, squarefree polynomials $(g_1, \ldots, g_d)$ such that $f = g_1 \cdot g_2^2 \cdot \ldots \cdot g_d^d$.

The next lemma shows that for monic polynomials the squarefree decomposition is unique. Moreover, this decomposition can be computed efficiently.

▶ **Lemma 23** (Theorem 14.23 of [48] and extensions). *Let $f(y) \in \mathbb{F}[y]$ be a non-constant, monic polynomial of degree at most d. Then there exists a unique squarefree decomposition into a sequence of monic polynomials. Moreover, there exists a deterministic algorithm that given the polynomial $f(y)$ uses $\text{poly}(d, \log|\mathbb{F}|)$ field operations and computes its squarefree decomposition.*

The squarefree decomposition gives rise to a simple e-th root computation algorithm for univariate polynomials. In addition, this algorithm can be used to test whether a univariate polynomial is indeed a perfect power.

▶ **Lemma 24.** *Let $g(y) \in \mathbb{F}[y]$ be a non-constant, monic polynomial of degree at most d an let $(g_1, \ldots, g_d)$ be its squarefree decomposition. Then $g(y) = h(y)^e$ for some $e \in \mathbb{N}$ and $h(y) \in \mathbb{F}[y]$ iff $g_i = 1$ when $e \not| i$.*

The proof can be found in Section D. The following is immediate given the previous lemmas.

▶ **Corollary 25.** *There exists a deterministic algorithm that given a non-constant, monic polynomial $f(y) \in \mathbb{F}[y]$ of degree at most d outputs a polynomial $h(y) \in \mathbb{F}[y]$ such that $f(y) = h(y)^e$ if one exists using* $\mathrm{poly}(d, \log|\mathbb{F}|)$ *field operations.*

We can extend the algorithm to handle arbitrary univariate polynomials by making a call to a root oracle.

▶ **Lemma 26.** *There exists a deterministic algorithm that given $e \in \mathbb{N}$, an e-th root oracle R_e and a polynomial $f(y) \in \mathbb{F}[y]$ of degree at most d uses* $\mathrm{poly}(d, \log|\mathbb{F}|)$ *field operations and one oracle call to R_e and computes an e-th root of $f(y)$. That is, the algorithm outputs a polynomial $h(y) \in \mathbb{F}[y]$ such that $f(y) = h(y)^e$ if one exists. Otherwise, the algorithm rejects.*

Proof. If $f(y) = \alpha \in \mathbb{F}$ is a field element (i.e. a constant polynomial), output $R_e(\alpha)$. Otherwise, consider $\hat{f}(y) \triangleq f(y)/\mathrm{lc}(f)$. As $\hat{f}(y)$ is a non-constant, monic polynomial we can apply Corollary 25 to compute $\hat{h}(y) \in \mathbb{F}[y]$ such that $\hat{f}(y) = \hat{h}(y)^e$. In addition, let $\alpha = R_e(\mathrm{lc}(f))$. Output $\alpha \cdot \hat{h}(y)$. Observing that $(\alpha \cdot \hat{h}(y))^e = f(y)$ completes the proof. ◀

3 Sparse Polynomials

In this section we discuss sparse polynomials, their properties and some related efficient algorithms which leverage these properties.

An *s-sparse polynomial* is polynomial with at most s (non-zero) monomials. We denote by $\|f\|$ the *sparsity* of f. In this section we list several results related to sparse polynomials. We begin with a corollary from [37] that shows that a sparse multilinear polynomial can be factored efficiently. Moreover, all its factors are sparse.

▶ **Lemma 27** ([37]). *Given a multilinear polynomial $f \in \mathbb{F}[x_1, x_2, \dots, x_n]$, there is a* $\mathrm{poly}(n, \|f\|)$ *time deterministic algorithm that outputs the* irreducible *factors, $h_1, \dots, h_k$ of f. Furthermore, $\|h_1\| \cdot \|h_2\| \cdot \dots \cdot \|h_k\| = \|f\|$.*

The following result gives an efficient reconstruction algorithm for sparse polynomials.

▶ **Lemma 28** ([29]). *Let $n, s, d \in \mathbb{N}$. There exists a deterministic algorithm that given an oracle access to an s-sparse polynomial $f \in \mathbb{F}[x_1, x_2, \dots, x_n]$ of degree d uses* $\mathrm{poly}(n, s, d, \log|\mathbb{F}|)$ *field operations and outputs f.*

As a corollary we obtain an efficient algorithm for testing identity and, more generally, similarity between sparse polynomials. We leave the proof of the corollary as an easy exercise for the reader.

▶ **Corollary 29.** *Let $f, g \in \mathbb{F}[x_1, x_2, \dots, x_n]$ be s-sparse polynomials of degree at most d. Then there exists an algorithm that given f, g uses* $\mathrm{poly}(n, d, s, \log|\mathbb{F}|)$ *field operations and tests if $f \sim g$. If yes, the algorithm also outputs $\alpha \in \mathbb{F}$ such that $f = \alpha g$.*

Additionally, we obtain an efficient algorithm for sparse polynomial division given an upper bound on the sparsity of the quotient polynomial. The main idea is to reconstruct to the quotient polynomial as a sparse polynomial, using the original polynomials as oracle access. Given a candidate sparse polynomial we then can verify whether it is indeed the quotient polynomial.

▶ **Lemma 30** ([29, 11]). *Let $n, s, d, t \in \mathbb{N}$. Let $f, g \in \mathbb{F}[x_1, x_2, \dots, x_n]$ be s-sparse polynomials of degree at most d. Then there exists an algorithm that given f, g uses* $\mathrm{poly}(n, d, s, t, \log|\mathbb{F}|)$ *field operations and computes the quotient polynomial of f and g if it a t-sparse polynomial. That is, if $f = gh$ for some $h \in \mathbb{F}[x_1, x_2, \dots, x_n]$, $\|h\| \le t$ then the algorithm outputs h. Otherwise, the algorithm rejects.*

Corollary 29 can be also extended to handle products of sparse polynomials.

▶ **Lemma 31** ([35]). *Let $n, s, d \in \mathbb{N}$. There exists a deterministic algorithm that given an oracle access to a product of s-sparse polynomials $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ when $f = \prod g_i$ of degree d uses* $\mathrm{poly}(n, s, d, \log|\mathbb{F}|)$ *field operations and tests if $f \equiv 0$.*

3.1 Sparse Multiquadratic Polynomials

In this section we prepare the ground for our main application - efficient factorization algorithm for sparse multiquadratic polynomials. We begin by showing that a factor of a sparse multiquadratic polynomials is also sparse. Recall that in general a sparse polynomial can have a dense factor.

▶ **Lemma 32.** *Let $0 \not\equiv f, g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be polynomials such that g is multiquadratic. Then $f \mid g \implies \|f\| \le \|g\|$.*

Proof. The proof is by induction on the number of variables. The base case is when $n = 0$. That is, $f, g \in \mathbb{F}$. Clearly, in this case $\|f\| = \|g\| = 1$ and the claim holds. Now suppose that $n \ge 1$. By definition, $f \cdot h = g$ for some $h \in \mathbb{F}[x_1, x_2, \ldots, x_n]$. We have two cases to consider: Suppose $\mathrm{var}(f) \cap \mathrm{var}(h) = \emptyset$. In this case $\|f\| \cdot \|h\| = \|g\|$ and hence $\|f\| \le \|g\|$. Otherwise, pick $x_i \in \mathrm{var}(f) \cap \mathrm{var}(h)$. Since g is multiquadratic we can write $f = f_i x_i + f_0$ and $h = h_i x_i + h_0$ such that f_i, h_i, f_0 and h_0 do not depend on x_i. Therefore: $\|g\| = \|(f_i x_i + f_0) \cdot (h_i x_i + h_0)\| = \|f_i h_i x_i^2 + (f_0 h_i + f_i h_0) x_i + f_0 h_0\| \ge \|f_i h_i\| + \|f_0 h_0\|$. By the induction hypothesis $\|f_i h_i\| \ge \|f_i\|$ and $\|f_0 h_0\| \ge \|f_0\|$. Consequently, $\|g\| \ge \|f_i h_i\| + \|f_0 h_0\| \ge \|f_i\| + \|f_0\| = \|f\|$ implying the claim of the lemma. ◀

It is easy to see that this bound is tight. The following corollary is immediate by combining the bound with Lemma 30.

▶ **Corollary 33.** *Let $n, s, d \in \mathbb{N}$. There exists an algorithm that given s-sparse multiquadratic polynomials $f, g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ uses* $\mathrm{poly}(n, s, d, \log|\mathbb{F}|)$ *field operations and computes the quotient polynomial of f and g. That is, if $f = gh$ for some $h \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ then the algorithm outputs h. Otherwise, the algorithm rejects.*

We can extend the result to the case when a polynomial is a factor of a product of sparse multiquadratic polynomials. Note that such a product need not be either sparse or multiquadratic.

▶ **Corollary 34.** *Let $0 \not\equiv f, g_1, \ldots, g_k \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be polynomials such that for all $i \in [k]$, g_i is multiquadratic. Then $f \mid g_1 \cdot \ldots \cdot g_k \implies \|f\| \le \|g_1\| \cdot \ldots \cdot \|g_k\|$.*

Proof. Since $f \mid g_1 \cdot \ldots \cdot g_k$, we can write $f = f_1 \cdot \ldots \cdot f_k$ such that $f_i \mid g_i$. By the Lemma: $\|f_i\| \le \|g_i\|$. Therefore: $\|f\| \le \|f_1\| \cdot \ldots \cdot \|f_k\| \le \|g_1\| \cdot \ldots \cdot \|g_k\|$. ◀

The following lemma shows that if a sparse multiquadratic polynomial over a field with an odd characteristic factors in a certain way, then the corresponding discriminant is a polynomial and, in fact, a sparse polynomial.

▶ **Lemma 35.** *Suppose* $\mathrm{char}(\mathbb{F}) \neq 2$. *Let $f = a x_i^2 + b x_i + c \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be a multiquadratic polynomial that can be factored as $f = g \cdot h$ when both g and h depend on x_i. Then there exists a multiquadratic polynomial $\Delta \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ such that $\Delta^2 = b^2 - 4ac$. Moreover, $\|\Delta\| \le \|f\|^2$.*

Proof. Let $g = g_i x_i + g_0$ and $h = h_i x_i + h_0$. By comparing the coefficients of x_i on both sides of the equation we get that $a = g_i h_i$, $b = g_i h_0 + g_0 h_i$ and $c = g_0 h_0$. Therefore, $b^2 - 4ac = (g_i h_0 + g_0 h_i)^2 - 4 g_i h_i g_0 h_0 = (g_i h_0 - g_0 h_i)^2$. Consequently, selecting $\Delta \triangleq g_i h_0 - g_0 h_i$ takes care of the first claim. The claim regarding the degree follows from the fact that the degree of every variable in $b^2 - 4ac$ is at most 4. Finally, as $(b + \Delta)(b - \Delta) = 4ac$, by Corollary 34: $\|b + \Delta\| \leq \|a\| \cdot \|c\|$, implying that $\|\Delta\| \leq \|a\| \cdot \|c\| + \|b\| \leq (\|a\| + \|b\| + \|c\|)^2 = \|f\|^2$. ◀

4 Computations beyond an Exponentiation Gate and Application

In this section we give all our results showing how perform certain computations on polynomials given an oracle access to their powers.

4.1 Evaluation beyond an Exponentiation Gate

The most basic task for polynomial manipulation is evaluating a polynomial given via an oracle access. In this section we show how to transform an oracle access to the polynomial f^e into an oracle access to f itself. This can be thought of having an oracle equipped with a clever root extraction algorithm. Our main result is given in the following algorithm.

```
Input: Oracle access to a polynomial $f = g^e \in \mathbb{F}[x_1, x_2, \ldots, x_n]$; $\bar{a} \in \mathbb{F}^n$ s.t.
       $f(\bar{a}) \neq 0$;
$e \in \mathbb{N}$, $e$-th root oracle $R_e$.
Evaluation points $\bar{b}_1, \bar{b}_2, \ldots \in \mathbb{F}[x_1, x_2, \ldots, x_n]$
Output: $h(\bar{b}_1), h(\bar{b}_2), \ldots$ when $h(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ is a polynomial s.t. $h^e = f$.

$\alpha \leftarrow R_e(f(\bar{a}))$ /* Computed only once.                 */
Compute $h_{\bar{b}}(t)$ such that $h_{\bar{b}}(t)^e = f(\ell_{\bar{a},\bar{b}}(t))$ /* Invoking Lemma 26   */
$\beta \leftarrow h_{\bar{b}}(0)$ ;
return $h_{\bar{b}}(1) \cdot \alpha / \beta$
```

Algorithm 1: Polynomial Oracle Transformation.

▶ **Lemma 36.** *Let $h(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be such that $f(\bar{x}) = h(\bar{x})^e$ and $h(\bar{a}) = \alpha$. Then for every $\bar{b} \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ Algorithm 1 outputs $h(\bar{b})$.*

Proof. First, by Lemma 15 such a polynomial $h(\bar{x})$ exists and is unique. In addition, $\beta \neq 0$ since $\beta^e = h_{\bar{b}}(0)^e = f(\ell_{\bar{a},\bar{b}}(0)) = f(\bar{a}) \neq 0$. Therefore, the output of algorithm is well-defined. Next, we have that $h_{\bar{b}}(t)^e = f(\ell_{\bar{a},\bar{b}}(t)) = h(\ell_{\bar{a},\bar{b}}(t))^e$. By Lemma 15, $h_{\bar{b}}(t) = \omega \cdot h(\ell_{\bar{a},\bar{b}}(t))$ for some $\omega \in \mathbb{F}$. Therefore: $\frac{h_{\bar{b}}(1)\cdot\alpha}{\beta} = \frac{\omega \cdot h(\ell_{\bar{a},\bar{b}}(1))\cdot\alpha}{h_{\bar{b}}(0)} = \frac{\omega \cdot h(\bar{b})\cdot\alpha}{\omega \cdot h(\bar{a})} = h(\bar{b})$. ◀

Note that Algorithm 1 requires a non-zero point of $f(\bar{x})$ as an additional input. Generally speaking, finding such a point is the well-known problem of Polynomial Identity Testing (PIT) which is not known to have an efficient deterministic algorithm. We now argue that for our purposes we do not need a PIT algorithm.

Recall that we are in the setting where the root of $f(\bar{x})$ is evaluated on a sequence of points. Given each new query point $\bar{b} \in \mathbb{F}^n$ we can first evaluate $f(\bar{x})$ on $\bar{b}$. If $f(\bar{b}) \neq 0$, we can set $\bar{a} = \bar{b}$ and use this $\bar{a}$ as the non-zero input onwards. Observe that Algorithm 1 works for the case $\bar{a} = \bar{b}$ as well. However, one may ask what happens with the previous query points? Or, what if for all the query points $\bar{b}$ are zeros of f? Observe that if $f(\bar{b}) = 0$ then

$h(\bar{b}) = 0$ for any $h(\bar{x}) \in \mathbb{F}[x_1, x_2, \dots, x_n]$ such that $h(\bar{x})^e = f(\bar{x})$. Therefore, there is no issue of inconsistency here and the oracle just needs to output 0. Consequently, we can patch Algorithm 1 by using the first non-zero query point as $\bar{a}$ (if one exists). Theorem 1 follows as a corollary of Lemma 36 and the above discussion.

4.2 Reconstruction beyond an Exponentiation Gate

An immediate application of the polynomial evaluation algorithm is reconstruction beyond an exponentiation gate. More formally, let A be a reconstruction algorithm for a circuit class $\mathcal{C}$. By definition, A requires an oracle access to $f \in \mathcal{C}$ to reconstruct it. We can extend the algorithm to reconstruct $f(\bar{x})$ given an oracle access to $f(\bar{x})^e$ and an e-th root oracle R_e, by simulating each query of A. However, in the spirit of Lemma 15 the reconstruction algorithm might end up outputting $\omega \cdot f(\bar{x})$ depending on the root oracle R_e at hand. This reasoning is summarized in Theorem 2. As a corollary we get the following:

Proof of Theorem 3. Apply Theorem 2 with Lemma 28. ◀

Theorem 4 also follows as a corollary given the following result:

▶ **Lemma 37** ([33]). *Let $n \in \mathbb{N}$. There exists a deterministic algorithm that given an oracle access to a read-once polynomial $f \in \mathbb{F}[x_1, x_2, \dots, x_n]$ uses* $\text{poly}(n) \cdot \text{poly}(\log |\mathbb{F}|)$ *field operations and outputs a read-once formula Ψ that computes f.*

4.3 Deterministic Factorization of Sparse Multiquadratic Polynomials

For the case of sparse multiquadratic polynomials we can actually push those techniques further to obtain complete factorization thus proving Theorem 6. We now give the overview of the algorithm. Suppose $\text{char}(\mathbb{F}) \neq 2$. Let $f \in \mathbb{F}[x_1, x_2, \dots, x_n]$ be a multiquadratic polynomial and let x_i be a variable such that f factors as $f = g \cdot h$ when both g and h depend on x_i. We can view f as $f = ax_i^2 + bx_i + c$ when $a(\bar{x}), b(\bar{x})$ are $c(\bar{x})$ polynomials that do not depend on x_i. Given this view, we can express g and h in terms of a, b and c using the quadratic formula. That is, we can write $a \cdot f = (ax_i + b/2 + \Delta/2) \cdot (ax_i + b/2 - \Delta/2)$ when Δ is a polynomial satisfying $\Delta^2 = b^2 - 4ac$. By Lemma 32, both factors are $\|f\|$-sparse so we could continue this process recursively. However, there are some issues with this approach. First, it is not clear that Δ is a polynomial since the expression $b^2 - 4ac$ might not be a perfect square. Next, suppose that Δ were a polynomial. Is it sparse? Answers to these question were given in Lemma 35. Finally, how do we compute Δ? For that purpose we apply Theorem 3 that allows us reconstruct a sparse polynomial f given an oracle access to its power f^e. Formally, an instantiation of Theorem 3 with $e = 2, d = 4n, s = \|f\|^2$ together with Lemma 35 give rise to the following corollary.

▶ **Corollary 38.** *Suppose* $\text{char}(\mathbb{F}) \neq 2$. *Let $f = ax_i^2 + bx_i + c \in \mathbb{F}[x_1, x_2, \dots, x_n]$ be a multiquadratic polynomial that can be factored as $f = g \cdot h$ when both g and h depend on x_i. Then there exists a deterministic algorithm that given $i \in [n]$, the polynomial $f(\bar{x})$ and a square root oracle R_2 uses* $\text{poly}(n, \|f\|, \log |\mathbb{F}|)$ *field operations and oracles calls, and outputs a multiquadratic polynomial $\Delta \in \mathbb{F}[x_1, x_2, \dots, x_n]$ such that $\Delta^2 = b^2 - 4ac$ and $\|\Delta\| \leq \|f\|^2$.*

However, this still does not solve the problem entirely, as we obtain a factorization of $a \cdot f$ instead of f, while a need not be constant. Another issue is that f could factor differently: $f = (a'x_i^2 + b'x_i + c')h$ and in particular the polynomial $a = a' \cdot h$ could be reducible. We solve both problems by changing the way we apply recursion: we first recursively factorize

$a(x)$ and then iteratively use Corollary 33 to write f as $f = \gcd(f, a) \cdot f'$. To finish the algorithm we need to observe that f' is either irreducible or factors as above. We now move the proof of Theorem 6.

```
Input: A multiquadratic polynomial $f(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$; A square root oracle $R_2$
Output: A list $h_1, \ldots, h_k$ of the irreducible factors of $f$. That is, $f = h_1 \cdot \ldots \cdot h_k$.

$\hat{f} \leftarrow \mathrm{lc}_{x_n}(f)$ ;
if $\hat{f}$ is a constant then $S \leftarrow \emptyset$ else $S \leftarrow \mathsf{Factor}(\hat{f})$;
$u \leftarrow f$; $T \leftarrow \emptyset$;
foreach $h \in S$ do
    $v \leftarrow u/h$; /* using the algorithm in Corollary 33. */
    if $v \neq \perp$ then $u \leftarrow v$ else $S \leftarrow S \setminus \{h\}$; $T \leftarrow T \cup \{h\}$;
end
if $\deg_{x_n}(u) = 1$ then
    return $S \cup \{u\}$
else
    Write $u = ax_n^2 + bx_n + c$ ;
    Compute $\Delta \leftarrow \sqrt{b^2 - 4ac}$; /* using the algorithm in Corollary 38. */
    $\eta_+ \leftarrow ax_n + b/2 + \Delta/2$; $\eta_- \leftarrow ax_n + b/2 - \Delta/2$;
    foreach $h \in T$ do
        $v \leftarrow \eta_+/h$; /* using the algorithm in Corollary 33. */
        if $v \neq \perp$ then $\eta_+ \leftarrow v$; else $\eta_- \leftarrow \eta_-/h$;
    end
    $\gamma \leftarrow \mathrm{lm}(u)/\mathrm{lm}(\eta_+ \cdot \eta_-)$;
    if $u = \gamma\eta_+ \cdot \eta_-$ then return $S \cup \{\gamma\eta_+, \eta_-\}$ else return $S \cup \{u\}$;
end
```

Algorithm 2: Factoring Sparse Multiquadratic Polynomials when $\mathrm{char}(\mathbb{F}) \neq 2$.

Proof of Theorem 6. The outline of the algorithm is given in Algorithm 2. First of all, as $f(\bar{x})$ is given to us as a list of monomials, we can assume wlog that $\mathrm{var}(f) = [n]$ by renaming the variables. The proof is by induction on $m(f) \triangleq |\mathrm{var}(\mathrm{lc}_{x_n}(f))|$.

Running time: Observe that throughout the execution of the algorithm $\|u\|, \|v\| \leq \|f\|$ and $\|\eta_+\|, \|\eta_-\| \leq \|f\|^2$. Initially, the bound holds by the definition of the polynomials. As each update results from a division, the claim regarding the sparsity follows from Lemma 32. Therefore, by Corollaries 33 and 38 we get that the total number of field operations and oracle calls to R_2 satisfies the following recurrent expression: $t(m, \|f\|) \leq t(m-1, \|f\|) + \mathrm{poly}(m, \|f\|, \log|\mathbb{F}|)$ resulting in $t(m, \|f\|) = \mathrm{poly}(m, \|f\|, \log|\mathbb{F}|)$. As $m \leq n - 1$, the claim regarding the running time follows.

Analysis: Suppose that $m(f) \geq 1$. We need to fix some notations. Let $f = h_1 \cdot \ldots \cdot h_k$ be a factorization of f into irreducible factors. Let g denote the product of those h_i-s that depend on x_n. Note that there can be at most two such factors. Therefore, we can write: $f = h_1 \cdot \ldots \cdot h_{k'} \cdot g$. Finally, let $\hat{g} = \mathrm{lc}_{x_n}(g)$ and let $\hat{g} = \hat{g}_1 \cdot \ldots \cdot \hat{g}_\ell$ be a factorization of $\hat{g}$ into irreducible factors. Note that $\gcd(g, \hat{g}) = 1$ since $x_n \notin \mathrm{var}(\hat{g})$ and

g contains only the factors the depend on x_n. Moreover, given the above we get that: $\hat{f} = h_1 \cdot \ldots \cdot h_{k'} \cdot \hat{g} = h_1 \cdot \ldots \cdot h_{k'} \cdot \hat{g}_1 \cdot \ldots \cdot \hat{g}_\ell$ is a factorization of $\hat{f}$ into irreducible factors. As $m(\hat{f}) < m(f)$, by the induction hypothesis the set S will contain the irreducible factors of $\hat{f}$. By the uniqueness of factorization, S will contain exactly the polynomials $\alpha_1 h_1, \ldots, \alpha_{k'} h_{k'}$ and $\beta_1 \hat{g}_1, \ldots, \beta_\ell \hat{g}_\ell$ for some $\{\alpha_i\}, \{\beta_j\} \subseteq \mathbb{F} \backslash \{0\}$. Consequently, the '**for each**' loop separates the h_i-s from $\hat{g}_j$-s by gradually dividing f by the containment of S. Observe, that at the end of the loop we get that: $S = \{\alpha_1 h_1, \ldots, \alpha_{k'} h_{k'}\}$, $T = \{\beta_1 \hat{g}_1, \ldots, \beta_\ell \hat{g}_\ell\}$. Moreover, as $u = f = h_1 \cdot \ldots \cdot h_{k'} \cdot g$ at the beginning of the loop and $\gcd(g, \hat{g}_j) = 1$ for every j, we get that $u = \frac{f}{\alpha_1 h_1 \cdot \ldots \cdot \alpha_{k'} h_{k'}} = \frac{g}{\gamma}$ for some $\gamma \in \mathbb{F}$. Therefore, to complete the algorithm we need to compute the irreducible factors of u and concatenate them with S. Recall that by definition g (and hence u) is a product of at most two irreducible polynomials, both depending on x_n.

If $\deg_{x_n}(u) = \deg_{x_n}(g) = 1$ then u must be a single irreducible factor and thus $f = \alpha_1 h_1 \cdot \ldots \cdot \alpha_{k'} h_{k'} \cdot u$ is a factorization of f into irreducible factors. Otherwise, $\deg_{x_n}(u) = \deg_{x_n}(g) = 2$ and there can be two cases. If u is irreducible, then again $f = \alpha_1 h_1 \cdot \ldots \cdot \alpha_{k'} h_{k'} \cdot u$ is a factorization of f into irreducible factors and the algorithm will return this factorization since for every η_- and η_+ the identity test $u =^? \gamma \eta_+ \cdot \eta_-$ will fail. Otherwise, we can write u as a product of two irreducible polynomials, both depending on x_n. By Corollary 38 the discriminant polynomial Δ in Line 12 is computed successfully. As $\gamma u = g$ we have that $\hat{g} = \gamma a$. Consequently, we can write $u \cdot \hat{g}_1 \cdot \ldots \cdot \hat{g}_\ell = u \cdot \hat{g} = u \cdot \gamma a = \gamma \eta_+ \cdot \eta_-$. As each $\hat{g}_i$ is an irreducible polynomial, it must be the case that either $\hat{g}_i \mid \eta_+$ or $\hat{g}_i \mid \eta_-$. Thus, at Line 17 we have that $u = \gamma \eta_+ \cdot \eta_-$. We can easily compute γ by noting that $\operatorname{lm}(u) = \operatorname{lm}(\gamma \eta_+ \cdot \eta_-) = \gamma \operatorname{lm}(\eta_+ \cdot \eta_-)$. In conclusion, $f = \alpha_1 h_1 \cdot \ldots \cdot \alpha_{k'} h_{k'} \cdot \gamma \eta_+ \cdot \eta_-$ is a factorization of f into irreducible factors and the algorithm will return this factorization passing the identity test $u =^? \gamma \eta_+ \cdot \eta_-$.

The analysis of the base case $m(f) = 0$ is similar. First, note that if $u = f$ is irreducible then the algorithm will return $\{u\}$. Otherwise, we can write u as a product of two irreducible polynomials, both depending on x_n. By definition, $a \cdot u = \eta_+ \cdot \eta_-$. As $a \neq 0 \in \mathbb{F}$, $\gamma = \frac{\operatorname{lm}(u)}{\operatorname{lm}(\eta_+ \cdot \eta_-)} = \frac{\operatorname{lm}(u)}{\operatorname{lm}(a \cdot u)} = \frac{1}{a}$ and hence $u = \frac{1}{a} \eta_+ \cdot \eta_- = \gamma \eta_+ \cdot \eta_-$. In conclusion we get that in the base case, $f = \gamma \eta_+ \cdot \eta_-$ is a factorization of f into irreducible factors and the algorithm will return this factorization passing the identity test $u =^? \gamma \eta_+ \cdot \eta_-$. This completes the proof. ◀

5 Discussion & Open Questions

In this paper we study computations beyond a (single) exponentiation gate and present some applications, with the main one being the first efficient deterministic factorization algorithm for sparse multiquadratic polynomials over odd characteristics. Can we devise such algorithms for multicubic polynomials? Or more generally, when the individual degree of each variable is constant? One of the milestones on the route to this goal has to do with estimating the sparsity of the factors of such polynomials. To this end, we propose the following conjecture:

▶ **Conjecture 39.** *There exists a function $\nu : \mathbb{N} \to \mathbb{N}$ such that if $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ is a polynomial with individual degrees at most d then $g \mid f \implies \|g\| \leq \|f\|^{\nu(d)}$.*

Our results show that $\nu(1) = \nu(2) = 1$. As we noted before, the value of $\nu(3)$ is unknown. We also note that the conjecture gives rise to an efficient deterministic algorithm for testing sparse factorization into polynomials with constant individual degrees.

In addition, combined with the randomized factorization algorithm of [49], we can obtain an efficient factorization algorithm for such polynomial. Using Theorem 7 we can this algorithm zero-error, Las Vegas algorithm (i.e. ZPP-type).

Another milestone in sparse polynomial factorization is computing a root of a sparse polynomial. Theorem 8 allows us to test whether the polynomial f is an e-th root of the polynomial g. But can we actually compute f given g? Once again, an upper bound on the corresponding sparsity could be useful. We can get the desired result by combining this bound with Theorem 3. We propose the following conjecture:

▶ **Conjecture 40.** *Suppose* $\mathrm{char}(\mathbb{F}) = 0$ *or "large enough". There exists a function* $\mu : \mathbb{N} \to \mathbb{N}$ *such that for for every* $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ *and* $e \in \mathbb{N}$*:* $\|f\| \leq \|f^e\|^{\mu(e)}$.

Note even when $n = 1$, there exist sparse-square polynomials. That is, polynomials f such that $\|f^2\| < \|f\|$, implying that $\mu(2) > 1$. For more details see [13, 9] and references within.

In addition, Example 6.1 in [44] shows that when the field characteristic is close to the degree of the polynomial in question, even a square root of sparse polynomial could be very dense. Therefore, the bound could only hold for "large enough" (in terms of n, d etc..) characterstic. Finally, can we extend Theorem 8 to fields with "small" characteristics? Perhaps, by extending Lemma 48?

Acknowledgments. The author would like to thank the anonymous referees for useful comments.

References

1 M. Agrawal, C. Saha, R. Saptharishi, and N. Saxena. Jacobian hits circuits: Hitting-sets, lower bounds for depth-d occur-k formulas & depth-3 transcendence degree-k circuits. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 599–614, 2012.

2 M. Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 67–75, 2008.

3 D. Angluin, L. Hellerstein, and M. Karpinski. Learning read-once formulas with queries. *J. ACM*, 40(1):185–210, 1993.

4 M. Beecken, J. Mittmann, and N. Saxena. Algebraic independence and blackbox identity testing. *Information & Computation*, 222:2–19, 2013. `doi:10.1016/j.ic.2012.10.004`.

5 M. Ben-Or and P. Tiwari. A deterministic algorithm for sparse multivariate polynominal interpolation. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 301–309, 1988.

6 D. Bshouty and N. H. Bshouty. On interpolating arithmetic read-once formulas with exponentiation. *JCSS*, 56(1):112–124, 1998.

7 N. H. Bshouty and R. Cleve. Interpolating arithmetic read-once formulas in parallel. *SIAM J. on Computing*, 27(2):401–413, 1998.

8 N. H. Bshouty, T. R. Hancock, and L. Hellerstein. Learning boolean read-once formulas with arbitrary symmetric and constant fan-in gates. *JCSS*, 50:521–542, 1995.

9 D. Coppersmith and J. Davenport. Polynomials whose powers are sparse. *Acta Arith.*, 58:79–87, 1991.

10 D. A. Cox, J. Little, and D. O'Shea. *Ideals, varieties, and algorithms – an introduction to computational algebraic geometry and commutative algebra (4. ed.)*. Undergraduate texts in mathematics. Springer, 2015.

11 Z. Dvir and R. Mendes de Oliveira. Factors of sparse polynomials are sparse. *CoRR*, abs/1404.4834, 2014.

12 Z. Dvir, A. Shpilka, and A. Yehudayoff. Hardness-randomness tradeoffs for bounded depth arithmetic circuits. *SIAM J. on Computing*, 39(4):1279–1293, 2009.

13 P. Erdös. On the number of terms of the square of a polynomial. *Nieuw Arch. Wisk*, 23:63–65, 1949.

14 S. Gao, E. Kaltofen, and A. G. B. Lauder. Deterministic distinct-degree factorization of polynomials over finite fields. *J. Symb. Comput.*, 38(6):1461–1470, 2004.

15 K. O. Geddes, S. R. Czapor, and G. Labahn. *Algorithms for computer algebra.* Kluwer, 1992.

16 A. Gupta, N. Kayal, and S. V. Lokam. Reconstruction of depth-4 multilinear circuits with top fanin 2. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 625–642, 2012. Full version at http://eccc.hpi-web.de/report/2011/153.

17 V. Guruswami and M. Sudan. Improved decoding of reed-solomon codes and algebraic-geometry codes. *IEEE Transactions on Information Theory*, 45(6):1757–1767, 1999.

18 T. R. Hancock and L. Hellerstein. Learning read-once formulas over fields and extended bases. In *Proceedings of the 4th Annual Workshop on Computational Learning Theory (COLT)*, pages 326–336, 1991.

19 V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

20 E. Kaltofen. Single-factor hensel lifting and its application to the straight-line complexity of certain polynomials. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC)*, pages 443–452, 1987. `doi:10.1145/28395.28443`.

21 E. Kaltofen. Factorization of polynomials given by straight-line programs. In S. Micali, editor, *Randomness in Computation*, volume 5 of *Advances in Computing Research*, pages 375–412. JAI Press Inc., Greenwhich, Connecticut, 1989.

22 E. Kaltofen. Polynomial factorization: a success story. In *ISSAC*, pages 3–4, 2003.

23 E. Kaltofen and B. M. Trager. Computing with polynomials given by black boxes for their evaluations: Greatest common divisors, factorization, separation of numerators and denominators. *J. of Symbolic Computation*, 9(3):301–320, 1990.

24 I. Kaplansky. *An Introduction to Differential Algebra.* Hermann, Paris, 1957.

25 M. Karchmer, N. Linial, I. Newman, M. E. Saks, and A. Wigderson. Combinatorial characterization of read-once formulae. *Discrete Mathematics*, 114(1-3):275–282, 1993.

26 Z. S. Karnin, P. Mukhopadhyay, A. Shpilka, and I. Volkovich. Deterministic identity testing of depth 4 multilinear circuits with bounded top fan-in. *SIAM J. on Computing*, 42(6):2114–2131, 2013.

27 N. Kayal. *Derandomizing some number-theoretic and algebraic algorithms.* PhD thesis, Indian Institute of Technology, Kanpur, India, 2007.

28 N. Kayal. An exponential lower bound for the sum of powers of bounded degree polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:81, 2012. URL: `https://eccc.weizmann.ac.il/report/2012/081/`.

29 A. Klivans and D. Spielman. Randomness efficient identity testing of multivariate polynomials. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 216–223, 2001.

30 S. Kopparty, S. Saraf, and A. Shpilka. Equivalence of polynomial identity testing and deterministic multivariate polynomial factorization. In *Proceedings of the 29th Annual IEEE Conference on Computational Complexity (CCC)*, pages 169–180, 2014. `doi:10.1109/CCC.2014.25`.

31 A. K. Lenstra, H. W. Lenstr, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen,*, 261(4):515–534, 1982.

32 R. J. Lipton and N. K. Vishnoi. Deterministic identity testing for multivariate polynomials. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 756–760, 2003.

33 D. Minahan and I. Volkovich. Complete derandomization of identity testing and reconstruction of read-once formulas. *Manuscript*, 2016. (submitted).

34 C. Saha, R. Saptharishi, and N. Saxena. A case of depth-3 identity testing, sparse factorization and duality. *Computational Complexity*, 22(1):39–69, 2013. doi:10.1007/s00037-012-0054-4.

35 S. Saraf and I. Volkovich. Blackbox identity testing for depth-4 multilinear circuits. *Combinatorica*, 2016. (accepted).

36 V. Shoup. A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic. In *ISSAC*, pages 14–21, 1991.

37 A. Shpilka and I. Volkovich. On the relation between polynomial identity testing and finding variable disjoint factors. In *Automata, Languages and Programming, 37th International Colloquium (ICALP)*, pages 408–419, 2010. Full version at http://eccc.hpi-web.de/report/2010/036.

38 A. Shpilka and I. Volkovich. On reconstruction and testing of read-once formulas. *Theory of Computing*, 10:465–514, 2014.

39 A. Shpilka and I. Volkovich. Read-once polynomial identity testing. *Computational Complexity*, 24(3):477–532, 2015.

40 A. Shpilka and A. Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5(3-4):207–388, 2010.

41 M. Sudan. Decoding of reed solomon codes beyond the error-correction bound. *Journal of Complexity*, 13(1):180–193, 1997.

42 M. Sudan, L. Trevisan, and S. P. Vadhan. Pseudorandom generators without the XOR lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001. doi:10.1006/jcss.2000.1730.

43 L. G. Valiant. Negation can be exponentially powerful. *Theoretical Computer Science*, 12(3):303–314, 1980.

44 I. Volkovich. Deterministically factoring sparse polynomials into multilinear factors and sums of univariate polynomials. In *APPROX-RANDOM*, pages 943–958, 2015.

45 I. Volkovich. Characterizing arithmetic read-once formulae. *ACM Transactions on Computation Theory (ToCT)*, 8(1):2, 2016. doi:10.1145/2858783.

46 I. Volkovich. A guide to learning arithmetic circuits. In *Proceedings of the 29th Conference on Learning Theory, (COLT)*, pages 1540–1561, 2016. URL: http://jmlr.org/proceedings/papers/v49/volkovich16.html.

47 J. von zur Gathen. Who was who in polynomial factorization. In *ISSAC*, page 2, 2006.

48 J. von zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge University Press, 1999.

49 J. von zur Gathen and E. Kaltofen. Factoring sparse multivariate polynomials. *Journal of Computer and System Sciences*, 31(2):265–287, 1985. doi:10.1016/0022-0000(85)90044-3.

A Sparse Polynomials with Constant Individual Degrees

In this section we present an efficient factorization testing algorithm for sparse polynomials with constant individual degrees. In particular, we prove Theorem 7. We begin by observing that a Subresultant (Definition 18) of two sparse polynomials with constant degrees is a (somewhat) sparse polynomial with a (slightly larger) constant degree.

▶ **Observation 41.** *Let $f, g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be s-sparse polynomials with individual degrees at most d. Then for every $i \in [n]$ and $j \leq d$ the polynomial $S_{x_i}(j, f, g)$ is an $s^{\mathcal{O}(d)}$-sparse polynomial with individual degrees at most $\mathcal{O}(d^2)$.*

▶ **Lemma 42.** *Let $f, g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be s-sparse polynomials a let d be a bound on the individual degrees of f. Then there exists an algorithm that given f and g tests if $g \mid f$ using* $\operatorname{poly}(n, s^d, \log|\mathbb{F}|)$ *field operations.*

Proof. For $i \in [n]$, let d_i and e_i denote the individual degrees of x_i in f and g, respectively. We can assume wlog that $\forall i : e_i \le d_i \le d$. Otherwise, the answer is, clearly, "no". The algorithm will follow the procedure outlined in Corollary 20: Output "yes" iff $\forall i$ with $e_i > 0$: $S_{x_i}(e_i - 1, f, g) \equiv 0$.
The correctness follows immediately from Corollary 20. The running time follows from Observation 41. ◀

The efficient division algorithm gives rise to an efficient procedure for computing GCD given a list of sparse irreducible polynomials. Theorem 7 follows as a corollary of this result.

▶ **Theorem 43.** *Let $f, g_1, \ldots g_m \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be s-sparse polynomials a let d be a bound on the individual degrees of f. More over, let $g = \prod g_i$ and suppose that g_i-s are irreducible. Then given $f, g_1, \ldots g_m$, Algorithm 3 computes* $\gcd(f, g)$ *using* $\operatorname{poly}(n, s^d, \log|\mathbb{F}|)$ *field operations.*

```
Input: $s$-sparse polynomials $f, g_1, \ldots g_m \in \mathbb{F}[x_1, x_2, \ldots, x_n]$
Output: $e_1, \ldots, e_m$ such that $\gcd(f, g) = \prod g_i^{e_i}$

Use Corollary 29 to collect similar polynomials /* wlog $g = \prod_{i=1}^{m'} g_i^{e'_i}$ and $e'_i \le d$,
    where $g_i$ are irreducible, pairwise coprime factors */
For each $i \in [m']$ find the maximal $e_i$, $0 \le e_i \le e'_i$ such that $g_i^{e_i} \mid f$. /* Using
    Lemma 42 */
```

Algorithm 3: Compute the GCD of sparse polynomials with constant individual degrees.

Proof. The claim regarding the running time follows from Corollary 29 and Lemma 42. Since g_i's are irreducible polynomials, there exist a subset S such that $g \sim \prod_{g_i \in S} g_i^{e'_i}$. Therefore, $\gcd(f, g)$ will be of the form $\prod_{g_i \in S} g_i^{e_i}$ for some $e_i \le e'_i$. As d is a bound on the individual degrees of f, we get that $e_i \le d$. ◀

B Deterministic Reconstruction Algorithm for Multilinear $\Sigma\Pi\Sigma\Pi(2)$ Circuits

In this section we prove Theorem 5. We build on the following result of Gupta et al. [16]:

▶ **Lemma 44** (Implicit in [16])**.** *Let $n, s \in \mathbb{N}$. Let A be an algorithm that given an oracle access to an s-sparse split polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ output its irreducible factors using $T(n, s)$ operations. Then there exists a deterministic algorithm that given an oracle access to the polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ computable by a multilinear $\Sigma\Pi\Sigma\Pi(2)$ circuit of size s and uses* $\operatorname{poly}(n, s, \log|\mathbb{F}|, T(n, s))$ *field operations and oracles calls to A, and outputs a $\Sigma\Pi\Sigma\Pi(2)$ circuit that computes f.*

Originally, they invoke the randomized black-box factorization algorithm of Kaltofen & Trager [23] along with Lemma 28 to obtain an efficient randomized reconstruction algorithm. We are able to derandomize the reconstruction algorithm by extending Algorithm 2 to handle

s-sparse split polynomials. These are polynomials that can be written as products of s-sparse (not necessarily irreducible) polynomials. Note that an s-sparse split polynomial need not be sparse. To this end, we require the following Folklore results. (See e.g. [48], [12], [40] and reference within).

▶ **Lemma 45** (Folklore). *Let $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be a polynomial of degree d and let $i \in [n]$. We can write: $f = \sum_{j=0}^{d} f_j \cdot x_i^j$ such that $\forall j, x_i \notin \mathrm{var}(f_j)$. Then there exists a deterministic algorithm that given i, j and an oracle access to f uses $\mathrm{poly}(n, d, \log |\mathbb{F}|)$ field operations and outputs an oracle for f_j.*

To handle a division of s-sparse split polynomials we will need a s-sparse version of Corollary 33. We give a somewhat stronger statement: a black-box version of Lemma 42.

▶ **Lemma 46.** *Let $n, s, d \in \mathbb{N}$. Let $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ be an s-sparse split polynomial with individual degrees at most d. There exists an algorithm that given an oracle access to f and an irreducible s-sparse polynomial $g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ with individual degrees at most d uses $\mathrm{poly}(n, s^{d^2}, \log |\mathbb{F}|)$ field operations and computes the quotient polynomial of f and g. That is, if $f = gh$ for some $h \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ then the algorithm outputs an oracle for h. Otherwise, the algorithm rejects.*

Proof. We can write $f = f' \cdot u$ where f' is the product of all the irreducible factors of f that depend on x_i. In addition, we can write $f' = \sum_{j=0}^{d} f'_j \cdot x_i^j$ such that $\forall j, x_i \notin \mathrm{var}(f'_j)$. Clearly, f' is s^d sparse and u is s-sparse split. Using Lemma 45, we can obtain oracles for $f'_j \cdot u$. For $i \in [n]$, let d_i and e_i denote the individual degrees of x_i in f and g, respectively. We can determine d_i using Lemma 31. Hence, we can assume wlog that $\forall i : e_i \leq d_i \leq d$. Otherwise, the answer is, clearly, "no". The algorithm will follow the procedure outlined in Corollary 20: Output "yes" iff $\forall i$ with $e_i > 0$: $S_{x_i}(e_i - 1, f, g) \equiv 0$ using Lemma 45 to perform the test. The correctness follows immediately from Corollary 20. For the running time, by Lemma 19, $S_{x_i}(e_i - 1, f, g) = S_{x_i}(e_i - 1, f', g) \cdot u^{e_i}$. $S_{x_i}(e_i - 1, f', g)$ is a determinant of a $(d_i - e_i + 2) \times (d_i - e_i + 2)$ matrix whose entries are s^d-sparse polynomials with individual degrees at most d resulting in an $s^{\mathcal{O}(d^2)}$-sparse polynomial with individual degrees at most $\mathcal{O}(d^3)$. Therefore, we can compute the expression using Lemmas 31 and 45. ◀

Based on the above we can now prove the s-sparse split version of Theorem 6.

▶ **Theorem 47.** *Let $n, s \in \mathbb{N}$ and suppose $\mathrm{char}(\mathbb{F}) \neq 2$. There exists a deterministic algorithm that given an oracle access to an s-sparse split multiquadratic polynomial $f(\bar{x}) \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and a square root oracle R_2 uses $\mathrm{poly}(n, s, \log |\mathbb{F}|)$ field operations and oracle calls to R_2 and outputs the irreducible factors of $f(\bar{x})$. That is, a list $h_1, \ldots, h_k$ of irreducible polynomials such that $f = h_1 \cdot \ldots \cdot h_k$.*

Proof. By definition, $f = h_1 \cdot \ldots \cdot h_k$. By Lemma 32, we can assume wlog that h_i-s are irreducible and are, in fact, the irreducible factors of f. Therefore, we can invoke Algorithm 2 with the following minor changes:

- In Line 1, use Lemma 45 to compute $\hat{f}$.
- In Line 5, use the algorithm of Lemma 46 with $d = 2$ instead of Corollary 33.
- In Line 11, use Lemma 28 to reconstruct u as an s^2-sparse polynomial.

The analysis of the algorithms essentially remains the same. Note that these change introduces only a polynomial overhead to sparsities of the intermediate polynomials (and thus to the algorithm). Yet, as was established above, the irreducible factors are s-sparse. ◀

Theorem 5 follows by applying Theorem 47 to Lemma 44.

C Polynomial Identity Testing beyond an Exponentiation Gate

Using techniques from Differential Field Theory we show how to transform an identity test of powers of polynomials into an identity test that involves partial derivatives of those same polynomials. This transformation can be applied for classes of polynomials that are closed under partial derivatives such as sparse polynomials.

▶ **Lemma 48.** *Let $f(\bar{x}), h(\bar{x}) \not\equiv 0 \in \mathbb{F}(x_1, x_2, \dots, x_n)$ and let $e, d \in \mathbb{N}$. There exists $c(\bar{x}) \in \mathbb{F}(x_1, x_2, \dots, x_n)$ such that $f(\bar{x})^d = c(\bar{x}) \cdot h(\bar{x})^e$ and $\frac{\partial c}{\partial x_i} \equiv 0$ iff $d \cdot h \cdot \frac{\partial f}{\partial x_i} = e \cdot f \cdot \frac{\partial h}{\partial x_i}$.*

Proof.
($\Rightarrow$) Suppose $f(\bar{x})^d = c(\bar{x}) \cdot h(\bar{x})^e$. Then $d \cdot h \cdot \frac{\partial f}{\partial x_i} = \frac{h}{f^{d-1}} \cdot \frac{\partial (f^d)}{\partial x_i} = \frac{h}{f^{d-1}} \cdot c(\bar{x}) \cdot e \cdot \frac{\partial h}{\partial x_i} \cdot h(\bar{x})^{e-1} = e \cdot c(\bar{x}) \cdot \frac{h(\bar{x})^e}{f^{d-1}} \cdot \frac{\partial h}{\partial x_i} = ef \cdot \frac{\partial h}{\partial x_i}$.
($\Leftarrow$) Consider $c \triangleq \frac{f^d}{h^e}$. By definition: $\frac{\partial c}{\partial x_i} = \frac{1}{h^{2e}} \cdot \left(d \cdot \frac{\partial f}{\partial x_i} \cdot f^{d-1} \cdot h^e - e \cdot \frac{\partial h}{\partial x_i} \cdot h^{e-1} \cdot f^d \right) = \frac{f^{d-1}}{h^{e+1}} \cdot \left(d \cdot \frac{\partial f}{\partial x_i} \cdot h - e \cdot \frac{\partial h}{\partial x_i} \cdot f \right) \equiv 0$ and the claim follows. ◀

The following theorem provides an algorithm for an identity testing of powers of polynomials over fields with zero or large enough characteristics.

▶ **Theorem 49.** *Let $f(\bar{x}), h(\bar{x}) \not\equiv 0 \in \mathbb{F}[x_1, x_2, \dots, x_n]$ be polynomials of degree at most δ and let $e, d \in \mathbb{N}$. Furthermore, suppose that $p \triangleq \operatorname{char}(\mathbb{F}) = 0$ or $p > \delta \cdot \min(e, d)$. Then $f(\bar{x})^d = h(\bar{x})^e$ iff $\operatorname{lm}(f)^d = \operatorname{lm}(h)^e$ and for each $i \in [n]$ we have that $d \cdot h \cdot \frac{\partial f}{\partial x_i} = e \cdot f \cdot \frac{\partial h}{\partial x_i}$.*

Proof.
($\Rightarrow$) Follows from Lemma 48 and the definition of lm.
($\Leftarrow$) By iterative application of Lemma 48 we get that there exists $c(\bar{x}) \in \mathrm{C}(\mathbb{F}(x_1, x_2, \dots, x_n))$ such that $f(\bar{x})^d = c(\bar{x}) \cdot h(\bar{x})^e$. We claim that $c(\bar{x}) \in \mathbb{F}$. Assume the contrary. Then, by Lemma 13 $p > 0$ and there exist $u(\bar{x}), v(\bar{x}) \in \mathbb{F}[x_1^p, x_2^p, \dots, x_n^p]$ such that $\gcd(u, v) = 1$ and $c(\bar{x}) = \frac{u(\bar{x})}{v(\bar{x})}$. Therefore, we can write: $f(\bar{x})^d \cdot v(\bar{x}) = h(\bar{x})^e \cdot u(\bar{x})$. By definition $\operatorname{lm}(f)^d \cdot \operatorname{lm}(v) = \operatorname{lm}(h)^e \cdot \operatorname{lm}(u)$, which implies that $\operatorname{lm}(v) = \operatorname{lm}(u)$. In particular, $v(\bar{x}), u(\bar{x}) \notin \mathbb{F}$ as $c(\bar{x}) \notin \mathbb{F}$ and thus $\deg(u), \deg(v) \geq p$. Assume wlog that $d \leq e$. Then $p > \delta d$. As $\gcd(u, v) = 1$ we get that $u \mid f^d$ which implies that $p \leq \delta d$ thus leading to a contradiction. Therefore, $c(\bar{x}) = \alpha \in \mathbb{F}$. By definition $\operatorname{lm}(f)^d = \alpha \cdot \operatorname{lm}(h)^e$, which implies that $\alpha = 1$ and we are done. ◀

Theorem 8 follows an as easy corollary by noting that the preconditions of Theorem 49 can be efficiently checked given two sparse polynomials. It is also to be noted that similar characterization could be obtained by considering the 2×2 Wronskian of the polynomials f^d and h^e. However, we believe that our proof is cleaner and more direct.

D Missing Proofs

Proof of Lemma 15.

1. If $h \equiv 0$ then clearly $g \equiv 0$ and the claim follows. Otherwise, let $h = h_1^{e_1} \cdot \ldots \cdot h_k^{e_k}$ and $g = g_1^{e'_1} \cdot \ldots \cdot g_{k'}^{e'_{k'}}$ be factorizations of h and g into irreducible, pairwise comprise factors, respectively. We have that $h_1^{e_1 \cdot e} \cdot \ldots \cdot h_k^{e_k \cdot e} = h^e = g^e = g_1^{e'_1 \cdot e} \cdot \ldots \cdot g_{k'}^{e'_{k'} \cdot e}$ are two factorizations of the same non-zero polynomial. By Lemma 14, $k = k'$ and, wlog $h_i \sim g_i$ and $e_i = e'_i$. Consequently, $h = \omega \cdot g$ for some $\omega \in \mathbb{F}$. Finally, $h^e = \omega^e \cdot g^e = \omega^e \cdot h^e$ and the claim follows.

2. First, note that $h(\bar{a})^e = f(\bar{a}) \neq 0$ and thus $h(\bar{a}) \neq 0$. Let us consider $u(\bar{x}) \overset{\Delta}{=} \frac{\alpha h(\bar{x})}{h(\bar{a})}$. By definition, $u(\bar{a}) = \frac{\alpha h(\bar{a})}{h(\bar{a})} = \alpha$ and $u(\bar{x})^e = \frac{\alpha^e h(\bar{x})^e}{h(\bar{a})^e} = \frac{f(\bar{a})f(\bar{x})}{f(\bar{a})} = f(\bar{x})$. Now, suppose there exists a polynomial $v(\bar{x}) \in \mathbb{F}[x_1, x_2, \dots, x_n]$ satisfying the same properties. By the first part of the Lemma we have that $u = \omega \cdot v$ for some $\omega \in \mathbb{F}$. Therefore, $\alpha = u(\bar{a}) = \omega \cdot v(\bar{a}) = \omega \cdot \alpha$ implying that $\omega = 1$. Consequently, $u = v$. ◀

Proof of Lemma 24. Let $(g_1, \dots, g_d)$ be as above. Consider the polynomial $h \overset{\Delta}{=} \prod_{e \mid i} g_i^{i/e}$. We have that: $h^e = \prod_{e \mid i} g_i^i = \prod_i g_i^i = g$ when the last equality follows from the property of g_i and we are done. For the other direction, let $g = h^e$ and let $(h_1, \dots, h_d)$ be the squarefree decomposition of $h(y)$. Consider the following sequence:

$$\hat{g}_i = \begin{cases} h_{i/e} & e \mid i \\ 1 & \text{otherwise} \end{cases}$$

We have that

$$\prod_i \hat{g}_i^i = \prod_{e \mid i} h_{i/e}^i = \prod_j h_j^{j \cdot e} = \left(\prod_j h_j^j \right)^e = h^e = g.$$

In addition, $(\hat{g}_1, \dots, \hat{g}_d)$ is a sequence of pairwise coprime, squarefree polynomials. By uniqueness, the sequence $(\hat{g}_1, \dots, \hat{g}_d)$ is squarefree decomposition of g and the claim follows. ◀

Communication Complexity of Statistical Distance[*][†]

Thomas Watson

University of Memphis, Memphis, TN, USA
Thomas.Watson@memphis.edu

Abstract

We prove nearly matching upper and lower bounds on the randomized communication complexity of the following problem: Alice and Bob are each given a probability distribution over n elements, and they wish to estimate within $\pm\epsilon$ the statistical (total variation) distance between their distributions. For some range of parameters, there is up to a $\log n$ factor gap between the upper and lower bounds, and we identify a barrier to using information complexity techniques to improve the lower bound in this case. We also prove a side result that we discovered along the way: the randomized communication complexity of n-bit Majority composed with n-bit Greater-Than is $\Theta(n \log n)$.

1998 ACM Subject Classification F.1.2 Modes of Computation

Keywords and phrases Communication, complexity, statistical, distance

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2017.49

1 Introduction

Statistical (a.k.a. total variation) distance is a standard measure of the distance between two probability distributions, and is ubiquitous in theoretical computer science. Expressing the distributions (over a universe of n elements) as vectors of probabilities $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$, the statistical distance is defined as

$$\begin{aligned}\Delta(x, y) \;\coloneqq\; \tfrac{1}{2}\textstyle\sum_{i\in[n]} |x_i - y_i| \;&=\; \max_{S\subseteq[n]}\big|\textstyle\sum_{i\in S} x_i - \sum_{i\in S} y_i\big| \\ &=\; \max_{S\subseteq[n]}\big(\textstyle\sum_{i\in S} x_i - \sum_{i\in S} y_i\big).\end{aligned}$$

This measure has various interpretations, such as the minimum over all couplings of the probability that the sample from x and the sample from y are unequal, or as twice the maximum advantage an observer can achieve in guessing whether a random sample came from x or from y (where x or y is used with probability 1/2 each).

Given its pervasiveness, it is natural to inquire about the computational complexity of estimating the statistical distance between two distributions x and y that are given as input. This topic has been studied before in several contexts:

- [25] showed that when each of x and y is succinctly represented by an algorithm that takes uniform random bits and produces a sample from that distribution (so our actual input is the description of this pair of algorithms), then (a decision version of) the problem of estimating $\Delta(x, y)$ is complete for the complexity class SZK (statistical zero knowledge). (For results about the complexity of other problems where the inputs are succinctly represented distributions, see [12, 13, 3, 14, 30, 29].)

[*] A full version of the paper is available at https://eccc.weizmann.ac.il/report/2016/170/.

[†] Supported by NSF grant CCF-1657377.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2017).
Editors: Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala; Article No. 49; pp. 49:1–49:10
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

- [2, 27, 9] studied the complexity of statistical distance estimation when an algorithm is only given black-box access to oracles that produce samples from the distributions specified by x and y. (For results about the complexity of other problems where the inputs are black-box samples from distributions, see the surveys [14, 24, 7].)
- [10, 11] studied the space complexity of (a generalization of) statistical distance estimation when the vectors x and y are provided as data streams.

1.1 Communication Upper and Lower Bounds

We study the statistical distance estimation problem in the context of communication complexity: Alice is given the vector x, Bob is given the vector y, and they wish to output a value in the range $\big[\Delta(x,y)-\epsilon, \Delta(x,y)+\epsilon\big]$. We let $\textsc{Stat-Dist}_{n,\epsilon}$ denote this two-party search problem. For any two-party search problem F, we let $\mathsf{R}(F)$ denote the minimum worst-case communication cost of any randomized protocol (allowing both public and private coins) such that for each input, the output is correct with probability at least 3/4. (For our problem $\textsc{Stat-Dist}_{n,\epsilon}$, the 3/4 can be replaced by any constant in the range $(1/2, 1)$ since we can amplify success probability by taking the median of multiple trials.) The following is a clean summary of our bounds.

▶ **Theorem 1.**

$$\mathsf{R}(\textsc{Stat-Dist}_{n,\epsilon}) \textit{ is } \begin{cases} \Theta(1/\epsilon^2) & \textit{if } 1 > \epsilon \geq 1/O(\sqrt{n}) \\ \Omega(n) \textit{ and } O(n\log n) & \textit{if } 1/\omega(\sqrt{n}) \geq \epsilon \geq 1/2^{o(n\log n)} \\ \Theta(\log(1/\epsilon)) & \textit{if } 1/2^{\Omega(n\log n)} \geq \epsilon > 0 \end{cases}.$$

We also go ahead and ascertain the deterministic communication complexity (denoted with D instead of R) of this problem. We prove Theorem 1 and Theorem 2 in Section 2.

▶ **Theorem 2.** $\mathsf{D}(\textsc{Stat-Dist}_{n,\epsilon}) = \Theta(n\log(1/\epsilon))$ *provided ϵ is at most a sufficiently small constant.*

Closing the gap in Theorem 1 is a principal open problem. We get slightly better bounds in certain narrow ranges of ϵ (see the proof), but e.g., it remains open to prove our conjecture that $\mathsf{R}(\textsc{Stat-Dist}_{n,1/2^n}) \geq \omega(n)$. A natural strategy is to use information complexity lower bound techniques; however, in the full version we exhibit a barrier to accomplishing this. Specifically, for a large class of inputs having a certain type of product structure (which arises naturally from attempts to use the direct sum property of information complexity), and for a wide range of ϵ, $\textsc{Stat-Dist}_{n,\epsilon}$ can be solved with $O(n)$ information cost and 0 error probability. This suggests that to improve the $\Omega(n)$ bound, we may need to look at inputs not having the aforementioned product structure, and we are at a loss for techniques in this case.

1.2 Composing with Majority

We take this opportunity to prove other results that we discovered in the process of trying to analyze $\textsc{Stat-Dist}_{n,\epsilon}$. Recall the famous direct sum conjecture stating that computing k independent copies of a two-party function should require $\Omega(k)$ times as much randomized communication as computing 1 copy. A somewhat stronger version of the conjecture states that even just computing the $\textsc{And}$ of k independent copies should still require $\Omega(k)$ times as much communication. [15] proved the query complexity analogue of this $\textsc{And}$-composition conjecture, as well as a communication complexity version that is weaker than the full conjecture in two senses: it is *qualitatively* weaker since instead of converting a protocol for And_k composed with F into a plain randomized (BPP-type) protocol for F with factor $\Omega(k)$ savings, the conversion results in a protocol in a slightly stronger model (which has been variously called 2WAPP [16, 15], two-sided smooth rectangle bound [18], and relaxed partition bound [19]); it is *quantitatively* weaker since besides the $\Omega(k)$ savings, the conversion incurs a logarithmic additive loss due to the use of the "information odometer" of [5]. (We provide the precise statement in Section 3.)

We prove that when composing with the k-bit Majority function Maj_k instead of And_k, the above quantitative deficiency can be avoided: we get a perfect $\Omega(k)$ factor savings by circumventing the need for the odometer (although we retain the qualitative deficiency). For the applications in [15, 1], the logarithmic additive loss in the And-composition result was immaterial albeit perhaps a slight nuisance. In some settings, however, that loss would be damaging; one such setting is the following corollary (which holds by combining our Maj-composition result with the lower bound of [6] for the Greater-Than function Gt_n on n-bit inputs).

▶ **Theorem 3.** $\mathsf{R}(\text{Maj}_n \circ \text{Gt}_n^n) = \Theta(n \log n)$.

Evaluating the function $\text{Maj}_n \circ \text{Gt}_n^n$ can be described by a story: Alice and Bob have taken some exams and know their own scores, and they wish to determine the victor of their rivalry: who got a higher score on the most exams?

We prove the Maj-composition result and provide details about Theorem 3 in Section 3. We make the stronger conjecture that Theorem 3 should hold even with And_n instead of Maj_n; this would follow from an $\Omega(\log n)$ information complexity lower bound for Gt_n with respect to a distribution only over 1-inputs (which is open but may be doable).

1.3 Preliminaries

We define And_n, Or_n, Maj_n as the And, Or, and Majority functions on n bits, and Eq_n, Gt_n, Disj_n, Gh_n as the Equality, Greater-Than, Set-Disjointness, and Gap-Hamming two-party functions where Alice and Bob each get n bits. We use $\mathbb{P}$ for probability, $\mathbb{E}$ for expectation, $\mathbb{H}$ for Shannon entropy, and $\mathbb{I}$ for mutual information. We generally use upper-case letters for random variables and corresponding lower-case letters for particular outcomes.

Randomized protocols by default have both public and private coins. We let $CC(\Pi)$ denote the worst-case communication cost of protocol Π. We let $IC_D(\Pi) := \mathbb{I}(T\,;\,X \mid Y, R) + \mathbb{I}(T\,;\,Y \mid X, R)$ denote the (internal) information cost with respect to (X, Y) sampled from the input distribution D, where the random variables T and R represent the communication transcript and public coins of Π, respectively.

2 Communication Upper and Lower Bounds

We now prove Theorem 1 and Theorem 2. As a preliminary technicality, we note that for the upper bounds, we may assume each of the probabilities x_i and y_i can be written exactly in binary with $\log(n/\epsilon) + O(1)$ bits. This is because if we truncate the binary representations to that many bits and reassign the lost probability to an arbitrary element in both x and y, this ensures at most $\epsilon/4$ mass has been shifted within each distribution, so their statistical distance changes by at most $\epsilon/2$; then to obtain an ϵ-estimation for the original x and y, we can run a protocol to get an $(\epsilon/2)$-estimation for the new x and y.

Proof of Theorem 1. In fact, we show that $\mathsf{R}(\textsc{Stat-Dist}_{n,\epsilon})$ is always

(i) $O(1/\epsilon^2)$,
(ii) $O(\max(n \log n, \log(1/\epsilon)))$,
(iii) $\Omega(\min(1/\epsilon^2, n))$,
(iv) $\Omega(\log(1/\epsilon))$,

which gives a slightly more detailed picture than the statement of Theorem 1.

The proof of (i) is inspired by the "correlated sampling lemma" that has been used in the context of parallel repetition [17, 22, 23] and earlier in the context of LP rounding [20]. As noted above, we may assume each probability x_i and y_i is a multiple of $1/m$ for some integer $m \coloneqq O(n/\epsilon)$. We make use of an $O(1)$-communication equality testing protocol that accepts with probability 1 when the inputs are equal and accepts with probability exactly 1/2 when the inputs are unequal (e.g., by using the inputs to index into a uniformly random public string and comparing the bits at those indices).

Here is the protocol witnessing (i). Alice and Bob repeat the following $O(1/\epsilon^2)$ times:

- Publicly sample a uniformly random ordering of $[n] \times [m]$.
- Alice finds the first $(i_\mathrm{A}, j_\mathrm{A})$ in the ordering such that $x_{i_\mathrm{A}} \geq j_\mathrm{A}/m$.
- Bob finds the first $(i_\mathrm{B}, j_\mathrm{B})$ in the ordering such that $y_{i_\mathrm{B}} \geq j_\mathrm{B}/m$.
- Run the equality test on $(i_\mathrm{A}, j_\mathrm{A})$ and $(i_\mathrm{B}, j_\mathrm{B})$.

Then they output $q/(1-q)$ where $q \coloneqq \min(1/2, \text{fraction of iterations where equality test rejected})$.

To analyze the correctness, let $\delta \coloneqq \Delta(x, y)$ and let p denote the probability the equality test rejects in a single iteration of the loop. We claim that $p = \delta/(1+\delta)$ (and hence $\delta = p/(1-p)$). To see this, define the following subsets of $[n] \times [m]$: $A \coloneqq \big\{(i,j) : x_i \geq j/m \text{ and } y_i < j/m\big\}$, $B \coloneqq \big\{(i,j) : x_i < j/m \text{ and } y_i \geq j/m\big\}$, and $C \coloneqq \big\{(i,j) : x_i \geq j/m \text{ and } y_i \geq j/m\big\}$. Then $|A| = |B| = \delta m$ and $|C| = (1-\delta)m$. The first (i^*, j^*) in the ordering to land in $A \cup B \cup C$ is uniformly distributed in that set. Thus with probability $\delta/(1+\delta)$ we have $(i^*, j^*) \in A$, in which case $(i_\mathrm{A}, j_\mathrm{A}) = (i^*, j^*) \neq (i_\mathrm{B}, j_\mathrm{B})$, and with probability $\delta/(1+\delta)$ we have $(i^*, j^*) \in B$, in which case $(i_\mathrm{A}, j_\mathrm{A}) \neq (i^*, j^*) = (i_\mathrm{B}, j_\mathrm{B})$, and with probability $(1-\delta)/(1+\delta)$ we have $(i^*, j^*) \in C$, in which case $(i_\mathrm{A}, j_\mathrm{A}) = (i^*, j^*) = (i_\mathrm{B}, j_\mathrm{B})$. It follows that the equality test rejects with probability $\frac{\delta}{1+\delta} \cdot \frac{1}{2} + \frac{\delta}{1+\delta} \cdot \frac{1}{2} + \frac{1-\delta}{1+\delta} \cdot 0 = \delta/(1+\delta)$.

By a Chernoff bound, the number of iterations guarantees that with probability at least 3/4, $|q - p| \leq \epsilon/8$. Since $\frac{d}{dp}\big[p/(1-p)\big] = 1/(1-p)^2 \in [1,4]$ for all $p \in [0, 1/2]$, it follows that $|\text{output} - \delta| = \big|q/(1-q) - p/(1-p)\big| \leq \epsilon/2$ whenever $|q-p| \leq \epsilon/8$ and $q \in [0, 1/2]$. This proves (i).

To prove (ii), we exploit the fact that the Greater-Than function $\textsc{Gt}_k$ with k-bit inputs can be computed with error probability $\gamma > 0$ and $O(\log(k/\gamma))$ bits of communication (by running the standard binary-search-based protocol [21, p. 170] for $O(\log(k/\gamma))$ many steps). As noted above, we may assume each probability x_i and y_i has $\log(n/\epsilon) + O(1)$ bits.

Here is the protocol witnessing (ii). For each $i \in [n]$, Alice and Bob compute $\textsc{Gt}(x_i, y_i)$ with error probability $1/(4n)$. Then Alice sends Bob the sum of x_i over all i for which the protocol for $\textsc{Gt}(x_i, y_i)$ accepted, and Bob sends Alice the sum of y_i over the same i's. They output Alice's sum minus Bob's sum. By a union bound, with probability at least 3/4 each of the $\textsc{Gt}$ tests returns the correct answer, in which case the final output is correct by definition. The communication cost is $O\big(n \log(n \log(n/\epsilon)) + \log(n/\epsilon)\big) \leq O(\max(n \log n, \log(1/\epsilon)))$.

To prove (iii), we use a reduction from the Gap-Hamming partial function $\textsc{Gh}_{n,\epsilon}$, in which the goal is to determine whether the relative Hamming distance between Alice's and Bob's length-n bit strings is $> 1/2 + \epsilon$ or $< 1/2 - \epsilon$. It is known that $\mathsf{R}(\textsc{Gh}_{n,\epsilon}) \geq \Omega(\min(1/\epsilon^2, n))$

[8, 28, 26]. Here is the reduction: Alice transforms $a \in \{0,1\}^n$ into a distribution x over $[2n]$ by letting $x_{2i-a_i} = 1/n$ for each $i \in [n]$ (and letting all other entries of x be 0). Bob transforms b into y in the same way. Then $\Delta(x,y)$ equals the relative Hamming distance between a and b, so a protocol for $\text{Stat-Dist}_{2n,\epsilon}$ can distinguish the two cases (by whether the output is above or below $1/2$).

To prove (iv), consider any correct randomized protocol for $\text{Stat-Dist}_{n,\epsilon}$, and fix any set of $1/(3\epsilon)$ many pairs of distributions having statistical distances 0, 3ϵ, 6ϵ, 9ϵ, ... There must exist some outcome of the randomness of the protocol such that the induced deterministic protocol is correct on at least three fourths of those inputs. But then the same transcript cannot occur for any two of these $1/(4\epsilon)$ inputs since the statistical distances are more than 2ϵ apart. Thus at least $1/(4\epsilon)$ transcripts are necessary, so the communication cost must be at least $\log(1/\epsilon) - 2$. ◀

Proof of Theorem 2. For the upper bound, assuming each probability x_i and y_i is a multiple of $1/m$ for some integer $m := O(n/\epsilon)$, we employ the trivial protocol where Alice sends a specification of her distribution to Bob (who then responds with the $(\log(n/\epsilon)+O(1))$-bit answer). We just need to count the number of such distributions: $\binom{m+n-1}{n-1} \leq \big(\frac{e\cdot(m+n-1)}{n-1}\big)^{n-1} \leq \big(O(1/\epsilon)\big)^n$. Hence only $O(n\log(1/\epsilon))$ bits are needed to specify a distribution.

The proof of the lower bound is basically a Gilbert–Varshamov argument for codes in the Manhattan metric. Specifically, we claim that there is a set of $2^{\Omega(n\log(1/\epsilon))}$ many distributions over $[n]$ that pairwise have statistical distance $> 2\epsilon$. Then for any distinct distributions x and x' from this set, the inputs (x,x) and (x',x') cannot share the same transcript in any correct protocol for $\text{Stat-Dist}_{n,\epsilon}$, because if they did then (x,x') would also share that transcript, but (x,x) requires output $\leq \epsilon$ while (x,x') requires output $> \epsilon$. Hence any correct protocol has at least $2^{\Omega(n\log(1/\epsilon))}$ transcripts and so has communication cost $\Omega(n\log(1/\epsilon))$.

To see the claim, first note that the number of distributions whose probabilities are multiples of $1/m$ is $\big(\Omega(1/\epsilon)\big)^n$, while the number of such distributions within statistical distance $\leq 2\epsilon$ of any fixed such distribution can be simply upper bounded by $2^n \cdot \binom{4\epsilon m+n}{n} \leq \big(O(1)\big)^n$. Hence if we keep greedily adding to a set any distribution that has statistical distance $> 2\epsilon$ from every distribution we picked so far, then the number of iterations this process can continue is at least $\big(\Omega(1/\epsilon)\big)^n / \big(O(1)\big)^n \geq \big(\Omega(1/\epsilon)\big)^n$, which is $2^{\Omega(n\log(1/\epsilon))}$ provided ϵ is at most a sufficiently small constant. ◀

3 Composing with Majority

In this section, we follow a convention that has become common in recent literature: For a two-party (possibly partial) function $F\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ and a complexity class name $\mathcal{C}$, we let $\mathcal{C}(F)$ denote the minimum worst-case cost of any protocol for F in the model corresponding to $\mathcal{C}$, and we also use $\mathcal{C}$ to denote the class of (families of) F's such that $\mathcal{C}(F) \leq \text{polylog}(n)$. In particular, $\mathsf{BPP}(F)$ is an alias for the plain randomized communication complexity $\mathsf{R}(F)$ in the case of $\{0,1\}$-valued F, but we use the complexity class notation now for aesthetic consistency. We also need the following "2-sided WAPP" model.[1]

[1] There are two ways to define this model, which are equivalent up to a factor of 2 in ϵ. Our way was also used in [16] and is the same as the relaxed partition bound [19]. In [15], a "starred" notation was used for this, while the notation 2WAPP was reserved for the other definition, which is the same as the two-sided smooth rectangle bound [18].

▶ **Definition 4.** $\mathsf{2WAPP}_\epsilon(F) := \min\big(CC(\Pi) + \log(1/\alpha)\big)$ over all $\alpha > 0$ and protocols Π with output values $\{0, 1, \perp\}$ such that for all (x, y), $\mathbb{P}[\Pi(x, y) \neq \perp] \leq \alpha$ and $\mathbb{P}[\Pi(x, y) = F(x, y)] \geq (1 - \epsilon)\alpha$.

For all F and constants $0 < \epsilon < 1/2$, we have $O(\mathsf{BPP}(F)) \geq \mathsf{2WAPP}_\epsilon(F) \geq \Omega(\mathsf{PP}(F))$, and thus $\mathsf{BPP} \subseteq \mathsf{2WAPP}_\epsilon \subseteq \mathsf{PP}$. It is not necessary to recall the communication complexity definition of PP, but we remark that $\mathsf{2WAPP}_\epsilon$ feels intuitively much closer to BPP, since there are many interesting classes sandwiched between $\mathsf{2WAPP}_\epsilon$ and PP [16]. The following is due to [16].

▶ **Theorem 5** (AND-composition). *For all F, k, and constants $0 < \epsilon < 1/2$, we have*

$$\mathsf{2WAPP}_\epsilon(F) \ \leq \ O\big(\mathsf{BPP}(\textsc{And}_k \circ F^k)/k + \log \mathsf{BPP}(\textsc{And}_k \circ F^k)\big).$$

We prove that by using $\textsc{Maj}_k$ instead of $\textsc{And}_k$, the logarithmic term can be avoided.

▶ **Theorem 6** (MAJ-composition). *For all F, k, and constants $0 < \epsilon < 1/2$, we have*

$$\mathsf{2WAPP}_\epsilon(F) \ \leq \ O\big(\mathsf{BPP}(\textsc{Maj}_k \circ F^k)/k + 1\big).$$

Proof of Theorem 3. As noted in the proof of Theorem 1, $\textsc{Gt}_n$ has a protocol with error probability $1/(4n)$ and communication cost $O(\log n)$. By running this on each of n coordinates, with probability at least $3/4$ all the outputs will be correct, so a protocol witnessing $\mathsf{BPP}(\textsc{Maj}_n \circ \textsc{Gt}_n^n) \leq O(n \log n)$ can be obtained by applying $\textsc{Maj}_n$ to all these outputs. The matching lower bound follows by combining Theorem 6 with the result that $\mathsf{PP}(\textsc{Gt}_n) \geq \Omega(\log n)$ [6]. ◀

Theorem 6 follows by stringing together the following three lemmas. For any input distribution D (over the domain of F), we define the distributions $D^b := (D \mid F^{-1}(b))$ for $b \in \{0, 1\}$. We say a protocol Π is δ-correct for F iff $\mathbb{P}[\Pi(x, y) = F(x, y)] \geq 1 - \delta$ for all (x, y).

▶ **Lemma 7.** *Fix any F, k, $0 < \delta < 1/2$, and input distribution D. For every δ-correct protocol Π for $\textsc{Maj}_k \circ F^k$ there exists a δ-correct protocol Π' for F such that $IC_{D^b}(\Pi') \leq O(CC(\Pi)/k)$ holds for both $b \in \{0, 1\}$.*

▶ **Lemma 8.** *Fix any F, input distribution D, and protocol Π (not necessarily correct). Then*

$$IC_D(\Pi) - 4 \ \leq \ \textstyle\sum_b \mathbb{P}_D[F^{-1}(b)] \cdot IC_{D^b}(\Pi) \ \leq \ IC_D(\Pi).$$

▶ **Lemma 9.** *Fix any F, constants $0 < \delta < \epsilon < 1/2$, and value c. If for every input distribution D there exists a δ-correct protocol Π for F such that $IC_D(\Pi) \leq c$, then $\mathsf{2WAPP}_\epsilon(F) \leq O(c + 1)$.*

Only the first inequality in Lemma 8 is needed for Theorem 6. Lemma 9 is due to [19]. Before we commence with the proofs of Lemma 7 and Lemma 8, we recall the following standard fact; see [4, §2.1] for a proof. (We apologize for overloading the D notation between this fact and the above lemmas, but there should be no confusion.)

▶ **Fact 10.** *Let A, B, C, D be four random variables. Then*
(i) $\mathbb{I}(A \,;\, B \mid C) \leq \mathbb{I}(A \,;\, B \mid C, D)$ *if* $\mathbb{I}(B \,;\, D \mid C) = 0$;
(ii) $\mathbb{I}(A \,;\, B \mid C) \geq \mathbb{I}(A \,;\, B \mid C, D)$ *if* $\mathbb{I}(B \,;\, D \mid A, C) = 0$.

Proof of Lemma 7. Assume k is odd for convenience. Consider a probability space with the following random variables: $Z \in \{0,1\}^k$ is a uniformly random string of Hamming weight $\lceil k/2 \rceil$, $S \coloneqq \{i : Z_i = 1\}$, (X, Y) is such that $(X_i, Y_i) \sim D^{Z_i}$ for each $i \in [k]$ independently, and T and R are the communication transcript and public coins (respectively) of Π on input (X, Y). We use the subscript notation $X_{<i}$ and $X_{>i}$ for restrictions to coordinates in $\{1, \ldots, i-1\}$ and $\{i+1, \ldots, k\}$, and we use the superscript notation X^S and X^{-S} for restrictions to coordinates in S and $[k] \smallsetminus S$, and we may combine these so e.g., $X^{-S}_{>i}$ is the restriction to coordinates in $\{i+1, \ldots, k\} \smallsetminus S$. We use corresponding notation for restrictions of Y. We have

$$\begin{aligned}
&2 \cdot CC(\Pi) \\
&\geq \mathbb{I}\big(T \,;\, X^S \,\big|\, X^{-S}, Y, R, S\big) + \mathbb{I}\big(T \,;\, Y^S \,\big|\, Y^{-S}, X, R, S\big) \\
&= \mathbb{E}_{s \sim S}\Big[\textstyle\sum_{i \in s} \mathbb{I}\big(T \,;\, X_i \,\big|\, X^s_{<i}, X^{-s}, Y, R, s\big) + \sum_{i \in s} \mathbb{I}\big(T \,;\, Y_i \,\big|\, Y^s_{>i}, Y^{-s}, X, R, s\big)\Big] \\
&\geq \mathbb{E}_{s \sim S}\Big[\textstyle\sum_{i \in s} \mathbb{I}\big(T \,;\, X_i \,\big|\, Y_i, X_{<i}, Y_{>i}, R, s\big) + \sum_{i \in s} \mathbb{I}\big(T \,;\, Y_i \,\big|\, X_i, Y_{>i}, X_{<i}, R, s\big)\Big] \\
&= \lceil k/2 \rceil \cdot \mathop{\mathbb{E}}_{\substack{s \sim S,\quad i \sim s,\quad r \sim R \\ x_{<i} \sim X_{<i},\quad y_{>i} \sim Y_{>i}}} \Big[\mathbb{I}\big(T \,;\, X_i \,\big|\, Y_i, x_{<i}, y_{>i}, r, s\big) + \mathbb{I}\big(T \,;\, Y_i \,\big|\, X_i, x_{<i}, y_{>i}, r, s\big)\Big]
\end{aligned}$$

where the second line is by the chain rule, the third line is by Fact 10.(i) since $X^{-s}_{>i}, Y_{<i}$ is independent of X_i given $Y_i, X_{<i}, Y_{>i}, R, s$ and since $Y^{-s}_{<i}, X_{>i}$ is independent of Y_i given $X_i, Y_{>i}, X_{<i}, R, s$, and where $i \sim s$ on the fourth line means i is sampled uniformly at random from the set s.

Note that sampling $s \sim S$ and $i \sim s$ is equivalent to sampling $i \sim [k]$ and a uniformly random balanced bit string $z_{-i} \sim Z_{-i}$ indexed by $[k] \smallsetminus \{i\}$ (and setting $z_i = 1$). We let $q \sim Q$ denote a sample of all the data $(i,\, z_{-i},\, r,\, x_{<i},\, y_{>i})$. In summary, we have

$$\mathbb{E}_{q \sim Q}\big[\mathbb{I}(T \,;\, X_i \,|\, Y_i, q) + \mathbb{I}(T \,;\, Y_i \,|\, X_i, q)\big] \;\leq\; (2/\lceil k/2 \rceil) \cdot CC(\Pi)$$

so by Markov's inequality, with probability $> 1/2$ over $q \sim Q$ we have

$$\mathbb{I}(T \,;\, X_i \,|\, Y_i, q) + \mathbb{I}(T \,;\, Y_i \,|\, X_i, q) \;\leq\; (4/\lceil k/2 \rceil) \cdot CC(\Pi) \tag{1}$$

where $(X_i, Y_i) \sim D^1$. By symmetric reasoning (interchanging the roles of 0 and 1), with probability $> 1/2$ over $q \sim Q$, (1) also holds if we instead have $(X_i, Y_i) \sim D^0$. Thus there exists a q (which we fix henceforth) such that (1) holds both when $(X_i, Y_i) \sim D^1$ and when $(X_i, Y_i) \sim D^0$ (and in either case, $(X_j, Y_j) \sim D^{z_j}$ for $j \neq i$).

Now consider the protocol Π' where the input is interpreted as (x_i, y_i), Alice privately samples $x_{>i} \sim (X_{>i} \,|\, y_{>i}, z_{>i})$, Bob privately samples $y_{<i} \sim (Y_{<i} \,|\, x_{<i}, z_{<i})$, and they run Π on the combined input (x, y) with public coins r. The conclusion of the previous paragraph is exactly that $IC_{D^b}(\Pi') \leq (4/\lceil k/2 \rceil) \cdot CC(\Pi) \leq O(CC(\Pi)/k)$ holds for both $b \in \{0, 1\}$. Furthermore, Π' is δ-correct since Π is δ-correct and $F(x_i, y_i) = (\textsc{Maj}_k \circ F^k)(x, y)$ with probability 1, for every (x_i, y_i) in F's domain. ◀

Proof of Lemma 8. Consider a probability space with the following random variables: $(X, Y) \sim D$, $F \coloneqq F(X, Y)$, and T and R are the communication transcript and public coins (respectively) of Π on input (X, Y). Then we have

$$\begin{aligned}
IC_D(\Pi) &= \mathbb{I}(T \,;\, X \,|\, Y, R) \quad\;\; + \mathbb{I}(T \,;\, Y \,|\, X, R) \\
\textstyle\sum_b \mathbb{P}_D[F^{-1}(b)] \cdot IC_{D^b}(\Pi) &= \mathbb{I}(T \,;\, X \,|\, Y, R, F) + \mathbb{I}(T \,;\, Y \,|\, X, R, F)
\end{aligned}$$

and so the second inequality of Lemma 8 holds by Fact 10.(ii) since conditioned on X, Y, R, there is no remaining entropy in F and hence it is independent of T.

For the first inequality, we use the following result proven in [15].

▶ **Lemma 11.** *There exist numbers $c_{x,y}, c'_{x,y} \geq 0$ for each input (x,y) in the domain of F, such that*

- $IC_D(\Pi) = \mathbb{E}[c_{X,Y}]$,
- $IC_{D^b}(\Pi) = \mathbb{E}[c'_{X,Y} \mid F = b]$ *for both* $b \in \{0,1\}$,
- *for each (x,y) in the domain of F, letting $b \coloneqq F(x,y)$ we have*

$$c_{x,y} \;\leq\; c'_{x,y} + \log\bigl(1/\mathbb{P}[F = b \mid y]\bigr) + \log\bigl(1/\mathbb{P}[F = b \mid x]\bigr).$$

Hence, letting $p_{x,y} \coloneqq \mathbb{P}[(X,Y) = (x,y)]$, we have

$$\begin{aligned} IC_D(\Pi) &= \textstyle\sum_{(x,y)} p_{x,y} \cdot c_{x,y} \\ &\leq \textstyle\sum_b \sum_{(x,y)\in F^{-1}(b)} p_{x,y} \cdot \bigl(c'_{x,y} + \log\bigl(1/\mathbb{P}[F = b \mid y]\bigr) + \log\bigl(1/\mathbb{P}[F = b \mid x]\bigr)\bigr) \\ &= \textstyle\sum_b \mathbb{P}[F = b] \cdot IC_{D^b}(\Pi) + \\ &\quad \textstyle\sum_b \sum_{(x,y)\in F^{-1}(b)} p_{x,y} \cdot \bigl(\log\bigl(1/\mathbb{P}[F = b \mid y]\bigr) + \log\bigl(1/\mathbb{P}[F = b \mid x]\bigr)\bigr). \end{aligned}$$

We claim that for both $b \in \{0,1\}$ we have $\sum_{(x,y)\in F^{-1}(b)} p_{x,y} \cdot \log\bigl(1/\mathbb{P}[F = b \mid y]\bigr) \leq 1$ and $\sum_{(x,y)\in F^{-1}(b)} p_{x,y} \cdot \log\bigl(1/\mathbb{P}[F = b \mid x]\bigr) \leq 1$; it then follows that $IC_D(\Pi) \leq \sum_b \mathbb{P}[F = b] \cdot IC_{D^b}(\Pi) + 4$.

We just argue the claim for $b = 1$ and conditioning on y; the other three cases are completely analogous. For $a \in \{0,1\}$ define $p^a_y \coloneqq \mathbb{P}[F = a \text{ and } Y = y] = \sum_{x\,:\,(x,y)\in F^{-1}(a)} p_{x,y}$. Then we have

$$\begin{aligned} \textstyle\sum_{(x,y)\in F^{-1}(1)} p_{x,y} \cdot \log\bigl(1/\mathbb{P}[F = 1 \mid y]\bigr) &= \textstyle\sum_y p^1_y \cdot \log\bigl((p^0_y + p^1_y)/p^1_y\bigr) \\ &\leq \textstyle\sum_y p^1_y \cdot \bigl((p^0_y + p^1_y)/p^1_y\bigr) \\ &= 1. \end{aligned}$$

This finishes the proof. ◀

Acknowledgements. I thank Mika Göös for discussions and anonymous reviewers for comments.

References

1 Anurag Anshu, Aleksandrs Belovs, Shalev Ben-David, Mika Göös, Rahul Jain, Robin Kothari, Troy Lee, and Miklos Santha. Separations in communication complexity using cheat sheets and information complexity. In *Proceedings of the 57th Symposium on Foundations of Computer Science (FOCS)*, pages 555–564. IEEE, 2016. doi:10.1109/FOCS.2016.66.

2 Tugkan Batu, Lance Fortnow, Ronitt Rubinfeld, Warren Smith, and Patrick White. Testing closeness of discrete distributions. *Journal of the ACM*, 60(1):4, 2013. doi:10.1145/2432622.2432626.

3 Andrej Bogdanov, Elchanan Mossel, and Salil Vadhan. The complexity of distinguishing Markov random fields. In *Proceedings of the 12th International Workshop on Randomization and Computation (RANDOM)*, pages 331–342. Springer, 2008. doi:10.1007/978-3-540-85363-3_27.

4 Mark Braverman. Interactive information complexity. *SIAM Journal on Computing*, 44(6):1698–1739, 2015. `doi:10.1137/130938517`.

5 Mark Braverman and Omri Weinstein. An interactive information odometer and applications. In *Proceedings of the 47th Symposium on Theory of Computing (STOC)*, pages 341–350. ACM, 2015. `doi:10.1145/2746539.2746548`.

6 Mark Braverman and Omri Weinstein. A discrepancy lower bound for information complexity. *Algorithmica*, 76(3):846–864, 2016. `doi:10.1007/s00453-015-0093-8`.

7 Clément Canonne. A survey on distribution testing: Your data is big. But is it blue? Technical Report TR15-063, Electronic Colloquium on Computational Complexity (ECCC), 2015. URL: `http://eccc.hpi-web.de/report/2015/063`.

8 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. `doi:10.1137/120861072`.

9 Siu On Chan, Ilias Diakonikolas, Paul Valiant, and Gregory Valiant. Optimal algorithms for testing closeness of discrete distributions. In *Proceedings of the 25th Symposium on Discrete Algorithms (SODA)*, pages 1193–1203. ACM-SIAM, 2014. `doi:10.1137/1.9781611973402.88`.

10 Joan Feigenbaum, Sampath Kannan, Martin Strauss, and Mahesh Viswanathan. An approximate L^1-difference algorithm for massive data streams. *SIAM Journal on Computing*, 32(1):131–151, 2002. `doi:10.1137/S0097539799361701`.

11 Jessica Fong and Martin Strauss. An approximate L^p-difference algorithm for massive data streams. *Discrete Mathematics & Theoretical Computer Science*, 4(2):301–322, 2001.

12 Oded Goldreich, Amit Sahai, and Salil Vadhan. Can statistical zero knowledge be made non-interactive? or On the relationship of SZK and NISZK. In *Proceedings of the 19th International Cryptology Conference (CRYPTO)*, pages 467–484. Springer, 1999. `doi:10.1007/3-540-48405-1_30`.

13 Oded Goldreich and Salil Vadhan. Comparing entropies in statistical zero-knowledge with applications to the structure of SZK. In *Proceedings of the 14th Conference on Computational Complexity (CCC)*, pages 54–73. IEEE, 1999. `doi:10.1109/CCC.1999.766262`.

14 Oded Goldreich and Salil Vadhan. On the complexity of computational problems regarding distributions. *Studies in Complexity and Cryptography*, pages 390–405, 2011. `doi:10.1007/978-3-642-22670-0_27`.

15 Mika Göös, T. S. Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication vs. partition number. In *Proceedings of the 44th International Colloquium on Automata, Languages, and Programming (ICALP)*. Schloss Dagstuhl, 2017. To appear.

16 Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016. `doi:10.1137/15M103145X`.

17 Thomas Holenstein. Parallel repetition: Simplification and the no-signaling case. *Theory of Computing*, 5(1):141–172, 2009. `doi:10.4086/toc.2009.v005a008`.

18 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. `doi:10.1109/CCC.2010.31`.

19 Iordanis Kerenidis, Sophie Laplante, Virginie Lerays, Jérémie Roland, and David Xiao. Lower bounds on information complexity via zero-communication protocols and applications. *SIAM Journal on Computing*, 44(5):1550–1572, 2015. `doi:10.1137/130928273`.

20 Jon Kleinberg and Éva Tardos. Approximation algorithms for classification problems with pairwise relationships: metric labeling and Markov random fields. *Journal of the ACM*, 49(5):616–639, 2002. `doi:10.1145/585265.585268`.

21 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1997.

22 Anup Rao. Parallel repetition in projection games and a concentration bound. *SIAM Journal on Computing*, 40(6):1871–1891, 2011. doi:10.1137/080734042.

23 Ran Raz. A counterexample to strong parallel repetition. *SIAM Journal on Computing*, 40(3):771–777, 2011. doi:10.1137/090747270.

24 Ronitt Rubinfeld. Taming big probability distributions. *ACM Crossroads*, 19(1):24–28, 2012. doi:10.1145/2331042.2331052.

25 Amit Sahai and Salil Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM*, 50(2):196–249, 2003. doi:10.1145/636865.636868.

26 Alexander Sherstov. The communication complexity of Gap Hamming Distance. *Theory of Computing*, 8(1):197–208, 2012. doi:10.4086/toc.2012.v008a008.

27 Paul Valiant. Testing symmetric properties of distributions. *SIAM Journal on Computing*, 40(6):1927–1968, 2011. doi:10.1137/080734066.

28 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. doi:10.4086/cjtcs.2012.001.

29 Thomas Watson. The complexity of deciding statistical properties of samplable distributions. *Theory of Computing*, 11:1–34, 2015. doi:10.4086/toc.2015.v011a001.

30 Thomas Watson. The complexity of estimating min-entropy. *Computational Complexity*, 25(1):153–175, 2016. doi:10.1007/s00037-014-0091-2.